Catching Spam Before It Arrives

by

Duncan Francis Cook, BComp

A dissertation submitted to the
School of Computing
in partial fulfilment of the requirements for the degree of

Bachelor of Computing with Honours

University of Tasmania

November, 2005

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma in any tertiary institution, and to my knowledge and belief, this thesis contains no material previously published or written by another person except where due reference is made in the text of the thesis.

……

Duncan Cook

Abstract

The arrival of any piece of unsolicited and unwanted email, commonly referred to as spam, is a source of annoyance to many email users. It results in real costs to individuals and organisations. Spam also contributes to a reluctance to use email by some individuals. Currently most spam prevention techniques rely on methods that examine the whole email message at the mail server. This thesis details research that aims to deny spam entry into the internal network, stopping it ever reaching the mail server. A system is described that can identify current potential spammer IP addresses in real-time and then inform all network gateways to block emails from those addresses. Various tests of the system's timeliness and efficiency are then illustrated, leading to a final conclusion of the system's viability and overall usefulness. This is followed by a discussion of various areas in which future research could be carried out.

Acknowledgements

To my wonderful supervisor Jacky, I can't thank you enough for the amount of help and support you provided over the course of the year. Your ability to remain enthusiastic about my work, even when my own enthusiasm had waned, motivated me to continue working and strive for further achievement. I am especially grateful of the encouragement I received when writing the conference paper, resulting in something I never expected to get out of this Honours year: an official academic publication.

To my co-supervisor Kevin, I am especially grateful for the amount of support you provided for my project, especially the access to live audit log data which essentially enabled this project to eventuate. Thank you for your willingness to meet in your own time after normal working hours to answer questions and provide valuable insight into network operations.

To Joel, the help and advice that you have given to me over the course of the year has been very much appreciated. The amount of input you had into my project could almost qualify you as my third supervisor.

To my brother Stuart, thank you for the invaluable technical help you provided in the implementation stages of my project. Your enthusiasm for programming continually amazes me.

To Mum and Dad, thanks for providing a roof over my head, food for me to eat and a car to get around in. Thank you for putting up with the fact that I did essentially no housework for those few tough months at the end of the year.

And finally, to all the honours people, thank you for making this my most enjoyable year at uni by a very long way. The beloved institutions of footy o'clock, magnet fishing and Magical Trevor kept me endlessly amused and made those long hours at uni just that bit more bearable. I am in awe of the sheer amount of talent exhibited by you all.

Table of Contents

Listing of Figures and Tables

Figures

Tables

1 Introduction

Unsolicited bulk email, commonly referred to as spam, is a major concern for the email infrastructure. Email is now a significant communications channel and could arguably be described as an essential form of communication in today's connected society. As the community so heavily relies on email, anything negatively affecting the functionality of the email infrastructure severely threatens its usefulness as a communications medium.

It is hard to determine the exact proportion of spam compared with regular email, with various reports listing figures of anywhere between 2.5% - 10% in 1998 (Cranor and LaMacchia 1998) to 60% in 2005 (MessageLabs 2005). While the spam message proportion is difficult to determine, there is no doubt that it is increasing, as the above figures show. Regardless of the specific percentage that spam accounts for, it is undoubtedly significant enough that large amounts of time and money are currently being spent to combat the rising tide of spam messages.

This thesis will begin by outlining the impact spam has on individuals, companies and the general perception of the usefulness of the email system. Next, the motivation for sending spam will be discussed, along with the techniques currently used by spammers. A new method of spam detection and prevention is then proposed, designed to reduce the amount of resources consumed by spam within a network. This is followed by an analysis of the viability, usefulness and effectiveness of the system. The thesis concludes with a summary combined with suggestions for future research within the area of network-based spam detection and prevention.

1.1 What is Spam?

The term "spam" has many definitions. A report published for the Australian Government by the National Office for the Information Economy (NOIE) defines spam as *"unsolicited electronic messages, usually transmitted to a large number of*

recipients" (2003, p. 6). Spam is also commonly referred to as Unsolicited Commercial Email (UCE) or Unsolicited Bulk Email (UBE). The same NOIE report also notes that spam messages share one or more of the following characteristics:

- They are sent in an untargeted and indiscriminate manner, often by automated means.
- They include or promote illegal or offensive content.
- Their purpose is fraudulent or otherwise deceptive.
- They collect or use personal information in breach of the *Privacy Act 1988* National Privacy Principles (NPPs).
- They are sent in a manner that disguises the originator.
- They do not offer a valid and functional address to which recipients may send messages opting out of receiving further unsolicited messages.

These characteristics are aimed at not only defining spam messages, but also to differentiate spam from legitimate bulk email. Bulk email is considered legitimate if the recipient has voluntarily had a previous business relationship with the sender and it can be reasonably assumed that the recipient would like to receive the message. Legitimate bulk email also includes any mailing lists the receiver has signed up for that provide an "opt-out" feature; allowing the user to remove their email address from the list at any time.

Spam is not restricted to the domain of email either. Spam can also be found in other places such as USENET groups and web logs (blogs). Recent reports suggest spam has moved to instant messaging services such as AOL Instant Messenger and MSN Messenger, creating a phenomenon known as *spim* (Paulson 2004). Spam has also spread to text-messaging services on mobile phones and personal digital assistants (PDAs). This is a concern because whereas the receipt of an email message generally bears little or no cost, recipients of mobile phone marketing may have to pay for the cost of the message (Pfleeger and Bloom 2005). Whilst these other methods of spam exist, this thesis is specifically focussed on dealing with email spam.

1.2 The Impact of Spam

Spam wastes time, money and resources at many levels of a networks' infrastructure. If spam messages make it all the way to a user's inbox, then manually removing this spam wastes the user's time and if that user is at work then it is also costing their employer money in lost productivity. Conservative estimates indicate that the total cost of spam on users (worldwide) in 2001 was €10 billion (approximately AU$16 billion) a year (Gauthronet and Drouard 2001). Other reports estimate that spam cost US companies alone $10 billion (approximately AU$13 billion) in lost productivity in 2003 (Bekker 2003).

Spam also consumes valuable computing resources. Every unsolicited email consumes bandwidth and network resources regardless of whether users actually receive it (Whitworth and Whitworth 2004). Spam messages cause delays for all Internet users as they waste resources on the Internet backbone, the fundamental infrastructure of the Internet. Furthermore, as some spammers use dictionary attacks and outdated address lists, many messages are rejected as being invalid by the receiving mail server and "bounced" back to the sender (Garcia et al. 2004), wasting even more Internet resources. Also, running any sort of spam filter on a mail server removes processing time from the server's major purpose: delivering email. It is worth noting that the people who are making money out of spam are generally not the people who bear the full impact of dealing with the spam messages.

A further cost of spam is its psychological effect on users regarding their willingness to communicate via email. According to a report published by the PEW Internet & American Life Project (Fallows 2003), 25% of email users surveyed admit that the ever-increasing volume of spam has reduced their overall use of email with 60% of those saying that it has reduced their email use in a significant way. Furthermore, 30% of email users are concerned that their filtering systems may block incoming legitimate messages and 23% are concerned that their emails to others may be blocked by filtering systems. These concerns initially seem rather conservative or perhaps even paranoid, but the truth is they are inevitably grounded in reality; customers of various Australian Internet Service Providers recently ended up having legitimate email blocked by Telstra's BigPond mail servers after their Internet

Service Providers were blacklisted by Telstra's spam filtering system (LeMay 2005). The numbers above also serve to further highlight the effect spam has on the public's perception of the usefulness of email. If the amount of spam saturating users' inboxes continues to increase, there is a very real threat that the general public will eventually abandon email for other, less frustrating methods of communication.

1.3 Spammer Motivation and Techniques

When the above impacts and costs of spam are taken into account, it is feasible to ask the question: *'Why does spam continue to exist?'* The answer to this is simply that no matter how strange a concept it may seem, spam gets results. A small percentage of email users actually do buy products advertised through spam emails. While the public perception of spam is largely negative, spammers would not be operating if it were not a viable source of income. Weiss (2003) notes that a spammer only needs to receive one hundred responses out of ten million spam messages (0.001% acceptance) to turn a profit. As the acceptance of offers contained in spam emails is generally proportional to the amount of people that the email is sent to, it makes economic sense for spammers to send the message to as many people as possible.

In order to reach a large volume of users, spammers require an equally large number of email addresses. These are usually collected in three different ways: by using programs known as spam-bots to scavenge for email addresses listed on web sites and message boards (particularly USENET groups), by performing a dictionary attack (pairing randomly generated usernames with known domain names to 'guess' a correct address) or by purchasing address lists from other individuals or organisations (Pfleeger and Bloom 2005).

Once they have addresses, spammers can use programs known as "bulk mailers" to automate the sending of spam. These programs can send huge volumes of email messages in a small amount of time. Some bulk mailing programs use open-relays (email servers that allow unauthorised users to send email) to send messages, effectively hiding the true address of the spammer. Bulk mailers can also fabricate the *from* address in email message headers to further hide the identity of the spammer (Garcia et al. 2004).

Another technique spammers utilise to send emails is with the use of *zombie networks*, also know as *bot networks*. Zombie is the term given to a computer that has been infected by a virus, worm, or Trojan Horse (Levy 2003), which allows remote entities to take control and use it for their own (usually illegal) purposes. A large amount of these computers, usually called a *network* or *army* can be co-opted to send spam emails, requiring little of the spammer's own computing power and network bandwidth. This technique is also popular as it protects the identity of the spammer (Paulson 2004).

2 Literature Review

The Introduction section discussed the effects of spam and analysed the reasons why people send spam. The next step is to discuss the techniques that are available to combat spam. This chapter looks in detail at various techniques and technologies currently employed to protect against the receipt of spam messages: where in the network these technologies can be applied and what action can be taken when a spam message has been identified. This section also provides some background information about the primary sources of data for this research; namely audit log files.

The most popular techniques currently employed to prevent the arrival of spam generally revolve around the use of filters. Filters examine various parts of an email message to determine whether or not it is spam. Filtering systems can be further classified based on the parts of the email messages that they use for spam detection. Origin or address-based filters typically use network information for spam classification, while content filters examine the actual contents of email messages. Sender authentication systems use network information in conjunction with changes to the email sending system in order to identify spam messages (Haskins and Nielsen 2005).

The two main places in the network infrastructure where spam detection and prevention technologies can operate are at the mail server and at individual users' computers (Haskins and Nielsen 2005). Filtering at the host (user) level allows individual users to tailor the filtering system to closely fit their personal definition of what constitutes a spam message. In contrast, spam filters that operate at the mail server have to offer a reasonable level of protection for all of the users they service, while simultaneously trying to produce the lowest amount of false positives in order to cater to all the users' different views on what constitutes spam. It must be remembered that one person's spam message could be another's interesting email. As regular network users generally do not get to provide input into spam filtering

policies at the mail server, it is important to respect their interests and try to ensure that the filter does not block any legitimate email or if this is not possible, only blocks a small proportion.

2.1 Taking Action

When an email has been classified as spam, either at the mail server or at an individual host, there are multiple ways that it can be dealt with. One solution is to simply delete the message so that it never gets to the user. This quickly frees up resources on the mail server or host machine, the downside of which is that users are not aware of exactly what has been deleted. Another solution is to redirect messages classified as spam to another, lower priority inbox. This process is known as sidelining and it protects the user from receiving spam directly into their primary inbox and also allows them the flexibility to periodically examine this low priority inbox to check for wrongly classified legitimate email. This method loses its effectiveness as the amount of spam a user receives increases, as this makes it more difficult to find legitimate emails in the spam filled lower priority inbox. A further downside to this approach is that the spam messages are still taking up resources on the mail server or host machine while they remain in the lower priority inbox (Haskins and Nielsen 2005). Other techniques include only censoring the main message, while still forwarding the email header onto the recipient. This allows the recipient or the network administrator to determine if a legitimate email has been blocked and has the opportunity to contact the originator of the blocked message.

2.2 Origin-Based Filters

Origin-based filters use information contained in the network headers of email messages to detect spam. IP and email address are the most common pieces of network information used. The three major types of origin-based filters are blacklists, whitelists and challenge/response systems.

2.2.1 Blacklists

Blacklists, also known as *realtime blackhole lists* (RBL) or *domain name system black lists* (DNSBL), can filter mail from mail servers or domains that have sent spam or are suspected of doing so. IP addresses of known or suspected spammers are entered into centrally maintained databases and made available as blacklists

through the Internet. Standard DNS lookups are used to query these databases at the time of Simple Mail Transfer Protocol (SMTP) connection or when mail is received, with spam classification occurring based on the reply given.

Blacklists are managed by various separate groups; each with its own focus and different policies in regards to how an IP address gets on (and off) the list (Allman 2003). Blacklist focus could include Request For Comments (RFC[1]) compliance, open relays, open proxies, IP addresses or domains that spam has actually come from or even dial-up users. RFC compliance refers to the checking of email messages to determine whether they comply with the various RFC documents listing email standards and conventions. An open relay is a mail server that does not require user authentication to send email, allowing anyone to send mail through the server. An open proxy is an unauthenticated proxy server that allows anyone to connect to it (and send spam). Dial-up users are also the focus of some blacklists, as most legitimate mail servers do not run over dial-up connections.

Blacklists also differ in how aggressive they are at categorizing spam sources; some lists prioritise avoiding false positives. A false positive is a legitimate email incorrectly classified as a spam message. Other more aggressive blacklists aim to catch the largest percentage of spam (which usually produces more false positives). Popular blacklists include Trend Micro's RBL+ (formally MAPS) (*Trend Micro RBL+ Service* 2005), the SPEWS list (*Spam Prevention Early Warning System* 2005) and the Spamhaus SBL and XBL lists (*The Spamhaus Project* 2005).

As blacklists only require DNS lookups, they have a very low CPU overhead and are generally easy to implement. Another advantage of blacklists is that they allow spam to be blocked at the SMTP connection phase, effectively stopping it from entering the network. Blacklists are not without disadvantages however, an example of such is the fact that they are maintained by an external entity. These lists could potentially be removed at any time without warning, leaving networks solely relying on these blacklists without any form of spam protection at all. A further downside of blacklists is that they are not as dynamic as other spam protection technologies,

[1] a collection of open Internet standards

making them easier for spammers who frequently change their IP address to avoid. Also, IP addresses that were once hijacked by a spammer could remain blocked for much longer than they need be, potentially leading to false positives. The effectiveness of a blacklist relies on the people who manage them; if blacklists are not updated in a timely manner, spam can get through.

Additionally, some blacklist providers (particularly SPEWS) neglect to specify the policies used to add and remove addresses from the list, effectively forcing network administrators who use these lists to trust in the judgement of other people. These blacklist providers could also potentially add addresses to their blacklist that the network needs to receive email from. Buisnesses could also be affected by this if they are blocked simply because their Internet Service Provider has a history of sending spam or if previous owners of their current IP space were spammers (Lueg 2004). There is no way to determine if this has taken place unless the network administrator can detect the lack of email coming from the specific address or domain. Blacklists can also be maintained locally, but they require a large amount of maintenance from the network administrator to retain their usefulness.

Another problem with blacklists is that as the amount of spam increases, the number of DNS lookups to check blacklists increases. Mail servers that use more than one blacklist are particularly affected by this. A study conducted at the Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory found that blacklist DNS lookups accounted for 14% of all their DNS lookups in 2004, up from less than 0.4% in 2000 (Jung and Sit 2004).

Spammers can circumvent blacklists to a certain degree by using *zombie networks* (as described in Section 1.3). As a *zombie network* comprises of many different computers, all of which could be from different domains, a blacklist on a specific domain would provide only minimal spam protection.

2.2.2 Whitelists

Whitelists allow users to specifically define “trusted” addresses that will immediately classify as legitimate all email received from those addresses (Pfleeger and Bloom 2005). Mail will only be accepted into the normal inbox if the sender exists on the

whitelist, so new contacts need to be added to the whitelist before effective correspondence can begin. An appealing quality of whitelists is that for most users a whitelist would be significantly smaller and easier to maintain than a blacklist. Also, mail flagged by a whitelist as legitimate can bypass further spam filters, effectively reducing the load on those filters.

A problem with whitelists, however, is that since the sender of email messages is not authenticated, spammers who can guess an address on the whitelist can then freely propagate spam to that address (Allman 2003). Additionally, if used by themselves, whitelists can tend to be overly restrictive as it is almost inevitable that legitimate mail will eventually be blocked or filtered into a lower priority mailbox. If this lower priority mailbox contains a large amount of spam, searching for valid messages could become a very difficult task. Whitelists are, therefore, best used when combined with other spam blocking techniques (Garcia et al. 2004).

2.2.3 Challenge/Response Systems

Challenge/response systems are an advanced version of whitelists, allowing senders who are not on the whitelist to have their emails received. Incoming messages from addresses not on the whitelist trigger an automatic reply (or *challenge*) to the sender, requiring them to prove that they are a real user and not an automated mailer. For example, the sender may be required to click on a link in the reply message and enter a valid email address and the ID number of the response message. If this process is completed, then the email successfully passes through the challenge/response system (Pfleeger and Bloom 2005).

The challenge/response method aims to protect against automated mailer programs by forcing the user to complete a task that is simple for a human but too complicated for a program to handle. Challenge/response systems also protect against spammers who manually send email, as the time required to complete the challenge could be better used sending spam to additional addresses. Challenge/response systems also help to protect against the generally large amount of false positives generated by traditional whitelist systems.

One problem with challenge/response systems is the issue of deadlock. If two parties who have never corresponded before both run challenge/response systems, the challenge sent by the recipient's system will be caught by the sender's challenge/response system and neither party will have the opportunity to provide an appropriate response (Barracuda Networks (Date Unknown)). This problem could be alleviated if the original sender adds the recipient's address to their whitelist (either manually or automatically) before commencing communication.

Another problem associated with the use of challenge/response systems is legitimate automated email lists that the user has subscribed to. These lists cannot respond to the challenge messages generated by the system, and mail from these sources may be marked as spam. As with the deadlock issue, this problem could be alleviated if the subscriber adds the mailing list address to their whitelist before subscribing to the list. Like whitelists, challenge/response systems are also as susceptible to a spammer guessing a whitelisted address, therefore allowing spam to get through.

2.3 Content Filters

While origin-based filters such as whitelists and blacklists examine email headers and other network information, content filters detect spam by looking inside the email and examining the message contents. Most content-based spam detection systems try to "understand" the text to various extents in order to identify spam (Allman 2003). A simple word filter, for example, could look to see if the message contains the words *Viagra* or *sex* or the phrases "*buy now"* or *"you've won"* to determine whether it is spam. Filters based on this technique are commonly called keyword-based filters. These filters can be highly context sensitive though, as a pharmacy may not want emails with the word *Viagra* in them filtered out. Popular content filter types include Bayesian filters and rule-based filters.

2.3.1 Bayesian Filters

Spam filtering systems using Naïve Bayesian classification were originally proposed by two separate parties at the AAAI-98 Workshop on Learning for Text Categorization; one was by Pantel and Lin (1998) and the other was by Microsoft Research (Sahami et al. 1998). Bayesian (also known as statistical) filters work by analysing the words inside an email message to calculate the probability that it is

spam. This probability is based on not only those words that provide evidence that a message is spam, but also on those words that provide evidence that a message is not spam. Words that are not generally found in spam messages contribute to the probability value in very much the same way as words that are frequently found in spam messages (Graham 2003).

To calculate an email's spam probability with a good degree of accuracy, Bayesian filters need to be "trained" by being given examples of what constitutes a spam email and what does not. The advantage of this technique is that, given appropriate time and training data, Bayesian filters can achieve a combination of extremely high accuracy rates with a low percentage of false positives (Graham 2003). The low amount of false positives generated by a Bayesian filter is useful, as users generally regard the classification of legitimate emails as spam as an order of magnitude worse than receiving spam incorrectly classified as legitimate (see Section 1.2). A further advantage of Bayesian filters is that they are constantly self-adapting. Provided they receive ongoing training data from the user, Bayesian filters evolve to stop new spam techniques.

The disadvantage of Bayesian filters is that, like all other content filters, they require the entire message to be received before analysis can begin. Furthermore, it follows that Bayesian filters are generally more CPU intensive than origin-based filters, as calculating Bayesian probabilities requires significantly more processing power than simply querying a list, as in systems such as blacklists and whitelists.

2.3.2 Rule-Based Filters

Rule-based, or heuristic, filters search the email message for patterns that indicate spam. These patterns could include specific words or phrases, malformed message headers and large amounts of exclamation marks and capital letters. Detection of a specific pattern attributes an amount of points to an email message and once the point value of an email exceeds a set threshold, it is classified as spam. Rule-based filters were the most common type of spam filter until 2002, when Bayesian filters became popular (Graham 2003).

The major disadvantage of rule-based filters is maintenance in general. In order to update the system to deal with the latest spammer techniques, new rule sets need to be obtained at regular intervals. Since the rule sets are largely static, they are easily defeated by spammer techniques such as word obfuscation. For example, a rule-based filter that checks for the word *Free* will not be able to detect the string *F*r*e*e* as spam. It is incredibly difficult to include rules for every possible misspelling of common spam words, which has limited the effectiveness of such filters.

2.4 Other Filters

While the above filters are the most commonly used spam filtering techniques, it is by no means an exhaustive list. Other data mining, machine learning and text classification techniques currently under research include: digest-based filters (Damiani et al. 2004), density-based filters (Yoshida et al. 2004), Chi-squared filters (O'Brien and Vogel 2003), global collaboration filters (Hulten et al. 2004) and artificial neural networks (Drewes 2002). Social network techniques, such as Reputation Network Analysis are also under investigation (Golbeck and Hendler 2004).

It should also be noted that the various spam filtering methods described are by no means mutually exclusive. A popular spam filtering program, SpamAssassin (*The Apache SpamAssassin Project* 2005), uses a combination of Bayesian filtering, rule-based filtering and blacklist checking to calculate a spam score for a particular email message. Messages that exceed a particular user-defined threshold are then marked as spam and dealt with appropriately. The advantage of this technique is that it can make use of the strengths of each spam filtering technology while also being less susceptible to each of the various technologies' weaknesses.

2.5 Sender Authentication Systems

In contrast to filtering systems, which examine various parts of email messages in order to detect spam, sender authentication systems operate on the idea that if the actual sender of an email message can be verified it will be easier to identify (and then block) spammers. According to Delaney (2005, p. 4), *"Creating email authentication is the first step to returning dispositional control of email to the*

recipient." The aim of sender authentication is to make senders of email more accountable. These systems force the identity of senders to be properly verified, thereby allowing people to block spammers more efficiently.

Sender authentication systems should also lower the effectiveness of email fraud systems, such as *phishing* scams and the Nigerian fraud (Costales and Flynt 2005). Phishing is the term used to describe when an apparently legitimate looking email is sent to try to deceive the recipient into revealing various personal details, usually account login information. Phishing works because victims believe the email has come from a legitimate institution that they have dealt with before. Sender authentication systems should make it harder for phishers as they will no longer be able to assume the email addresses of the legitimate institutions their messages purport to be from (Delany 2005). Sender authentication systems have the added benefit of immediately blocking spam with no sender identification or if sender identification has been forged. Different types of sender authentication systems include reverse DNS, the Sender Policy Framework, the Sender ID Framework and DomainKeys.

2.5.1 Reverse DNS/ Reverse MX

Reverse DNS (also known as reverse MX or reverse mail exchange) systems were developed to combat the problem of email IP address spoofing. SMTP contains no facilities to authenticate the true sender of the message. To combat this, Danisch (2004) describes a system to validate sender identity without having to resort to using processor expensive cryptography. Reverse DNS and reverse MX combat spam by querying the DNS server of the sender's mail transfer agent and running a check to determine if the IP address of the incoming SMTP connection is authorized to send messages from the mail transfer agent. Pfleeger and Bloom comment that pure Reverse DNS systems tend to be unreliable because IP addresses can map to multiple domains (2005). AT&T WorldNet deployed a reverse DNS system in 2003 but had to remove it only 24 hours later when it was found that a great deal of legitimate mail was being blocked (Olsen 2003).

2.5.2 The Sender Policy Framework

The Sender Policy Framework (SPF) is another proposed solution to address spoofing and builds upon Danisch's Reverse MX system (2004) and the Designated Mailers Protocol proposed by Gordon Fecyk (2003). SPF was designed by Meng Wong of POBOX[2] and uses reverse MX information, published in the DNS record of a domain, to determine whether the mail exchange sending the email is authorised to send mail from the domain. SPF examines the *mail from:* parameter of the incoming message and can also examine the *HELO* (or *EHLO*) parameter from the sending SMTP server to determine if the message has come from an authorized sender in the domain (Wong and Schlitt 2004). Figure 2.1 shows an example of what a SPF record would look like in a DNS lookup:

```
v=spf1 +mx a:colo.example.com/28 -all
```

Figure 2.1: Sample SPF record (Grote 2004)

The above DNS lookup record comprises of four pieces of information. The *v=spf1* denotes the SPF version being used, in this case version 1, *+mx* denotes the listing is a valid mail exchange, *a:colo.example.com/28* identifies the reverse lookup address of the authorized outbound mail server and the *–all* specifies that if the domains do not match, reject the message as a forgery. If a message from another domain is received that lists its source address as example.com, the receiving mail server will deemed to have been forged and be rejected at the SMTP connection attempt (Grote 2004). It is important to note that SPF is dependent on DNS lookups, and is therefore only as secure as the DNS itself. This is somewhat of an issue, as vulnerabilities of the DNS root servers have recently been exploited (Baranowski 2003).

2.5.3 The Sender ID Framework

The Sender ID Framework is Microsoft's attempt at defining a standard for SMTP message authentication. It is heavily based on Wong's SPF (see Section 2.5.2) and also draws on Lyon's Purported Responsible Address proposal (2004). Sender ID employs SPF records in order to determine if a specific email message has been

[2] http://spf.pobox.com

received from an authorized host. Sender ID proposes the concept of 'positional modifiers' in the SPF records published in a DNS, which address some weaknesses in the SPF specification by modifying the outcome of the SPF *check_host()* function to examine the scope of the mail server in question (Lyon and Wong 2004). Figure 2.2 shows the steps in the Sender ID Framework authentication process:

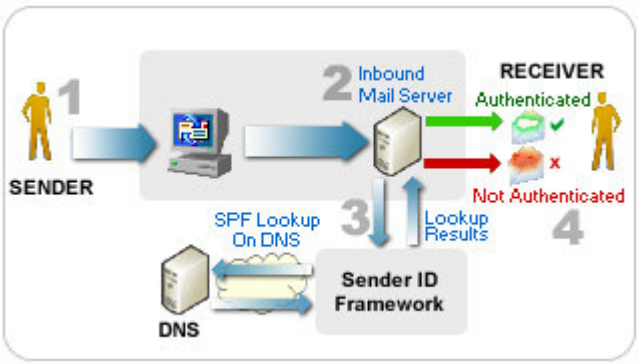

Figure 2.2: Sender ID operation (Microsoft Corporation 2005)

The following are the major steps in the Sender ID authorisation:

1. The sender transmits an email message to the receiver's mail server
2. The receivers mail server receives the message over SMTP
3. The receiver's inbound mail server queries the DNS of the sender's outbound mail server and checks to see if there is an SPF record that matches the address recorded in the SMTP session.
4. If the addresses match, the sender of the message is authenticated and the mail is made available to the recipient.

(Microsoft Corporation 2005)

The Sender ID Framework initially came under criticism from the open source community (including the Debian Project and the Apache Software Foundation), as the license to use it was not compatible with the GPL[3]. This prompted the authors of the original Sender ID Internet-Draft to refine the original specifications proposals and re-submit them to the IETF (Internet Engineering Task Force) for comment and

[3] The open source General Public License

consensus (Wagner 2004). There has been no word of late from Microsoft regarding the current state of this proposal.

2.5.4 DomainKeys

The DomainKeys approach to sender authentication utilises public/private key cryptography in association with DNS to verify both the identity of the sender and the integrity of the received email message. In order to implement the DomainKeys system, a domain owner must create one or more public/private keypairs that will be used for digitally signing valid emails sent from that domain. The domain owner places the public key in the DNS record associated with the domain and makes the private key available to all authorized mail servers in the domain. When a message is to be sent, the following takes place (Delany 2005):

1. The outbound mail server digitally signs the message with its private key. The digital signature of the message is then stored inside the header of the message and the message is sent to the recipient's mail server. The default signature is an RSA signed SHA-1 digest of the entire email (including headers).
2. The inbound mail server receives the message and extracts the signature and the claimed domain from the message headers.
3. The inbound mail server queries the DNS of the claimed domain for its public key.
4. The inbound mail server uses the sending domain's public key to verify if the message was signed with the corresponding private key, and thus, whether the message was sent with the authorization of the claimed sending domain. This will also verify that the message has not been altered whilst in transit.

The DomainKeys system only aims at authenticating the sender at the domain level; it provides no assurance that a given message comes from the purported user inside that domain. The responsibility of authenticating users within the domain rests with the outgoing mail server, which should only sign a message with its private key if it can verify the identity of the message originator.

As the DNS of the sending mail server is only used for public key storage and retrieval, the DomainKeys system can address some of the shortfalls in the SPF specification. While the DomainKeys specification still recommends that the signing services be deployed at the email boundary of an organisation (after all possible message header modification has been completed), two canonicalization algorithms are proposed to combat the fact that some components of the message header may be modified by internal relay services after the message has been signed. These algorithms can allow for some minor modification of message headers whilst still verifying integrity of the message.

One drawback with using public/private key cryptography to authenticate sender domain and message integrity is that it is more computationally expensive than using a reverse MX system like SPF or the Sender ID Framework. This increased processing time could become a major issue for email providers who process large amounts of mail. A further issue to be considered is that the mail servers need to be trusted to be secure enough to keep their private key secret. Also, these servers need to be set up to only send mail from authorised users within the domain.

2.6 Audit Log Analysis

In order to create a system that can detect and respond to spam attacks in real-time, there needs to be a facility for the gateways and mail servers and even hosts to record what is happening at any given time. This facility is provided by the use of *audit logs*. Audit logs have long been used in computer security to detect intrusions; Clifford Stoll, for example, used system printouts to detect and track a hacker intruding into the Berkeley University computer network (1991).

Audit log information can come from a variety of sources, including operating systems, firewalls, routers, mail servers and third party software (Amoroso 1999). As all these sources can produce a large amount of information, choosing the right system events to audit is of crucial importance. There is always a trade-off between the amount of information a system collects (how thorough the audit log will be) and the amount of system overhead the logging processes and stored log files use (Kemmerer and Vigna 2002). Generating too much log data makes it harder to analyse all the information, while not generating enough data may lead to an

ineffective system. The primary sources of audit log data used in this research project are *syslog* files, generated by the Linux kernel firewall, the *PortSentry* intrusion detection system and the *Sendmail* mail transfer agent.

2.6.1 Syslog

The UNIX system logger (syslog) is a commonly used logging application and it provides the log sources for this research. Syslog provides a way for different processes, applications and devices to send log information to a centralised point, known as the syslog server (Lonvick 2001). The three distinct sources that syslog events come from are: processes running on the local machine (the machine running the syslog daemon), kernel routines running on the local machine and processes running on other machines. All messages contain the source of the message, the authorizations associated with the message, the priority assigned to the message and the content of the message. For every message sent to the syslog server, a timestamp and message type keyword is appended, along with a new line character at the end. The syslog.conf file is consulted and the message is then handled in one of the following ways: sent to a file or specific UNIX device, sent to a user (e.g. root) or all users, sent to a program using the UNIX *pipe* command or sent to another machine (Amoroso 1999). Table 2.1 shows five lines (sanitised) from a Syslog file while Table 2.2 explains what the various lines mean.

1	Aug 1 04:42:01 ns1 portsentry[15060]: attackalert: SYN/Normal scan from host: 192.168.1.1/192.168.1.1 to TCP port: 135
2	Aug 1 04:42:01 ns1 portsentry[15060]: attackalert: Host: 192.168.1.1/192.168.1.1 is already blocked Ignoring
3	Aug 1 04:42:02 ns1 kernel: Packet log: input DENY eth0 PROTO=6 192.168.1.2:4285 192.168.1.3:135 L=48 S=0x00 I=18255 F=0x4000 T=108 SYN (#145)
4	Aug 1 04:43:14 ns1 portsentry[15060]: attackalert: SYN/Normal scan from host: 192.168.1.4/192.168.1.4 to TCP port: 135
5	Aug 1 04:43:14 ns1 portsentry[15060]: attackalert: Host 192.168.1.4 has been blocked via dropped route using command: "/sbin/ipchains -I input -s 192.168.1.4 -j DENY -l"

Table 2.1: Syslog file extract

1	PortSentry has detected a SYN/Normal scan from IP 192.168.1.1 to TCP port 135
2	Another scan from the same address was detected. Since it was blocked, it is simply a log entry that another probe was detected
3	The kernel firewall has detected a scan from address 192.168.1.2 to port 135. This packet is denied entry to the network
4	PortSentry has detected a SYN/Normal scan from IP 192.168.1.4 to TCP port 135
5	PortSentry has created an ipchains command to DENY packets from 192.168.1.4 entry to the network

Table 2.2: Syslog entries explained

2.6.2 Sendmail

The mail server logs used in this research came from the Sendmail Mail Transfer Agent. The Sendmail audit log entries are also written in syslog format, with a timestamp and message type keyword (in this case "sendmail") appended before the main Sendmail line. Table 2.3 shows three lines (sanitised) from a Sendmail log file, with the sending mail transfer agent's IP address emphasised.

1	Aug 1 04:43:28 ns1 sendmail[27026]: j6VIhPP27026: from=<hlsbing@spamdomain.de>, size=1305, class=0, nrcpts=1, msgid=<200507311843.j6VIhPP27026@example.com>, bodytype=8BITMIME, proto=SMTP, daemon=MTA, relay=c-67-171-253-79.ms1.example.com [192.168.1.1]
2	Aug 1 05:25:32 ns1 sendmail[28999]: j6VJPSP28999: from=<LLIU4@RCN.COM>, size=812, class=0, nrcpts=1, msgid=<4344461122848585@fny94-3-82-225-104-67.fbx.eg.net>, proto=ESMTP, daemon=MTA, relay=fny94-3-82-225-104-67.fbx.eg.net [192.168.1.2]
3	Aug 1 05:37:32 ns1 sendmail[29475]: j6VJbJP29475: from=<>, size=2158, class=0, nrcpts=1, msgid=<4c9d01c59601$7a89c406$a9403e31@ex2.net>, proto=SMTP, daemon=MTA, relay=host197.200-43-176.isp.net.ar [192.168.1.3]

Table 2.3: Example Sendmail log lines

This review has shown that there are many different issues to be considered when examining techniques to protect against spam. Current spam prevention methods can operate at the host or mail filter level. When running spam protection at the mail filter, a balance needs to be found that combines user satisfaction (low false positive rates) with actual spam protection (low false positives). One of the problems with current systems is that there is no communication between spam filters of different mail servers within a single network, allowing spammers to reuse the same attacks against multiple mail servers in the same domain. This presents a new avenue for

research: protection against spam through collaboration between multiple network gateways and mail servers.

3 Previous Work

Christopher MacLeavy, as part of his 2004 Honours work, analysed the combined audit logs for all gateways within a single class C IP address. SMTP (Sendmail) log files were compared with Linux syslog files in order to investigate if any precursor network activity could be detected on the network before the arrival of a spam message. The steps MacLeavy carried out in this investigation are detailed below and shown in Figure 3.1:

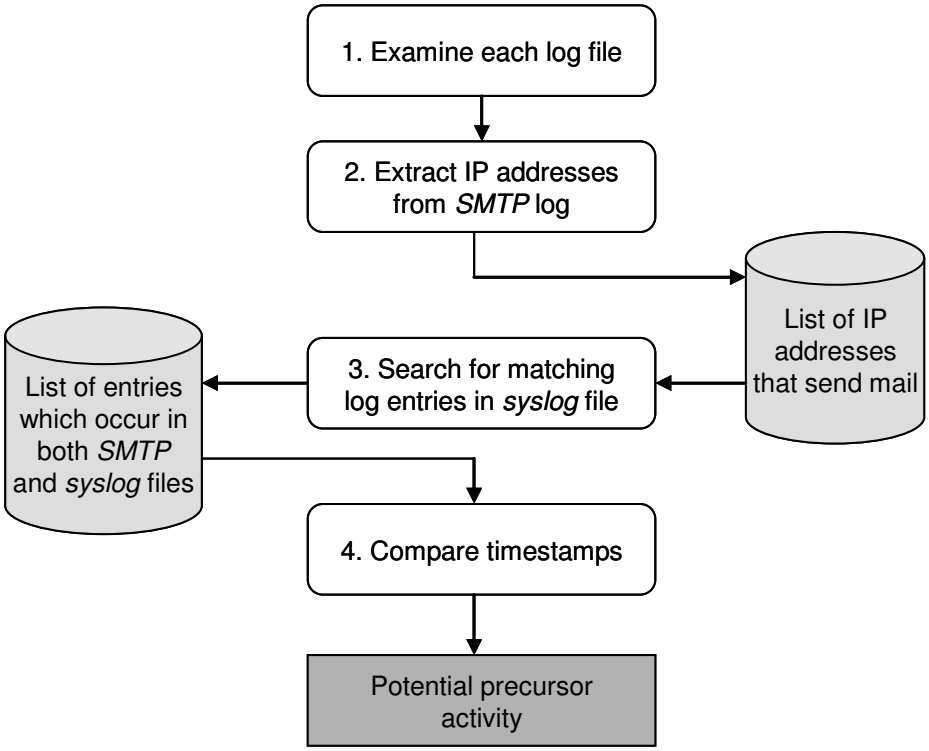

Figure 3.1: MacLeavy system operation (Adapted from MacLeavy (2004))

1. Both log file types were examined in order to determine the structure of the data and the different components inside
2. The IP address from the *relay* field of each email message received was extracted and stored in a file
3. For each IP address contained in the newly-created file, each line of the syslog file was searched for any entries containing a matching source address

4. For each syslog line flagged in the previous step, the timestamp was extracted and compared to the subsequent mail entry from the Sendmail log. Entries from the syslog that occurred after the mail was received were discarded. All other syslog entries still remaining in the result set were now considered potential precursor activity to the sending of spam.

MacLeavy examined the output file and determined that precursor activity was occurring, though the proportion of spam with precursors compared to spam without precursors was not reported. MacLeavy found that most precursor activity observed took the form of a probe to TCP port 0. He strengthened this assertion by stating *"There is no known motivation for a machine probing a receiving network before sending e-mail other than to ensure the safe arrival of spam."* It was also found that 63% of mail with a precursor arrives at the mail server less than sixty seconds after the aforementioned precursor. Furthermore, 89% of spam arrives at the gateway less than 10 minutes after its precursor.

4 Methodology

This thesis examines network precursors to spam, continuing the work started by Chris MacLeavy in 2004. Specifically, the aim of this thesis is to determine whether it is feasible to build and operate a system that can identify spam precursors in real-time and dynamically adapt to block spam packets at the network gateway. This prevents these spam packets from entering the network at all, effectively reducing the load placed upon various internal network resources, most importantly lowering the amount of email messages that the mail server has to process. With this aim in mind, the first task was to validate the results presented by Chris MacLeavy in his 2004 Honours thesis. The second task was to investigate whether a system can be created that can operate in real-time, detecting spam precursors and blocking spam messages access to the network.

4.1 Multiple Gateway Protection

One of the goals of this research was to provide spam protection across multiple network gateways in real-time. Large networks often contain multiple computers that allow computers inside the network to access other connected networks or the Internet. These computers also allow traffic from outside the network, for example email messages, to enter the network and proceed to their intended destination (in the case of email, this destination is the network's mail server). The computers that facilitate this connection between the inside and the outside of the network are known as gateways. Large networks may also contain multiple mail servers. The methods proposed in this thesis aim to provide correlation between all of a network's mail servers in order to identify spam attacks that are occurring across one or more mail server, possibly coming in to the network through different gateways. Once an attack is confirmed and the offending IP address identified, all network gateways need to be notified to block all traffic originating from this IP address from entering the network. It must be noted that this IP is the address of the mail relay (mail transfer agent) that the spam originated from, and all subsequent references to

"Spammer IP" actually refers to the IP of the spamming relay. Figure 4.1 shows a diagram of a simple network with three gateways connected to two mail servers.

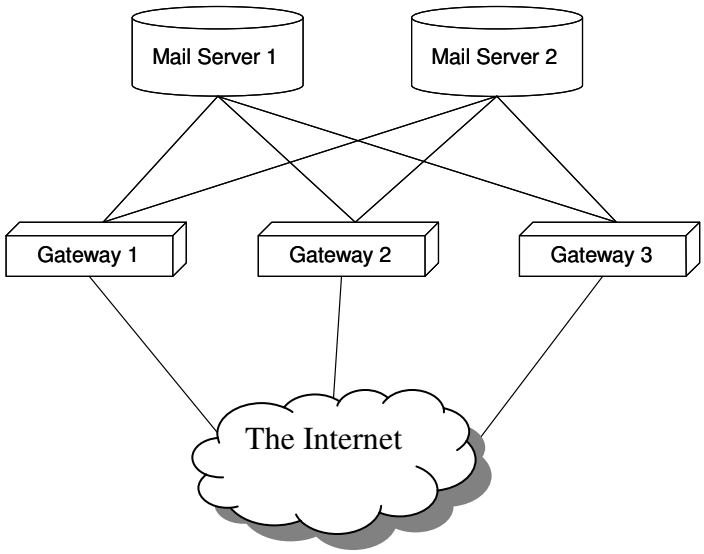

Figure 4.1: Simplified network diagram

With current forms of spam protection, all email messages that arrive at the gateways are forwarded to the network's mail servers. The mail servers then provide the first line of protection against spam messages and can filter them before they reach individual users. The new spam protection system proposed in this research aims to stop spam messages at the network gateways, preventing them from entering the network.

4.2 Phase 1 – Validation

Validation of the MacLeavy system was an important step as the aims for this thesis are based on the results of MacLeavy's work. This phase was also needed due to the use of "live" audit log data in this research. As some of the audit log files contained almost 300 MB of text (approximately two million lines) to cover a 21 day time period, they were deemed too large to sort through manually. Therefore, a system was needed to reduce the logs to a manageable size and in the process, identify the sort of network activity to be examined in the real-time system. The existing implementation of the MacLeavy system was deemed too inefficient to use for this validation exercise, as it consumed approximately 52% of the CPU resources of a Sun Sparc Ultra Enterprise 250 server (with dual 400 MHz UltraSparc-II processors and 2GB of memory) for the majority of 8 days (MacLeavy 2004, p. 32). This

resulted in the need for a new system to be developed based on MacLeavy's methodology that could output results in significantly less time on the machine provided for this research: an 800MHz Pentium 3 with a total of 512MB of memory running the Ubuntu Linux operating system.

4.2.1 Phase 1 System Overview

The new system (hereafter referred to as the Phase 1 system) was developed in the *Perl* scripting language, using Perl's powerful regular expression engine to traverse an amalgamated audit log spanning multiple gateways and mail servers. Perl's regular expression engine is ideally suited to large text-processing operations, allowing meaningful data to be easily extracted for large data sources by using pattern matching against regular expressions. Figure 4.2 shows an overview of the operation of the Phase 1 system:

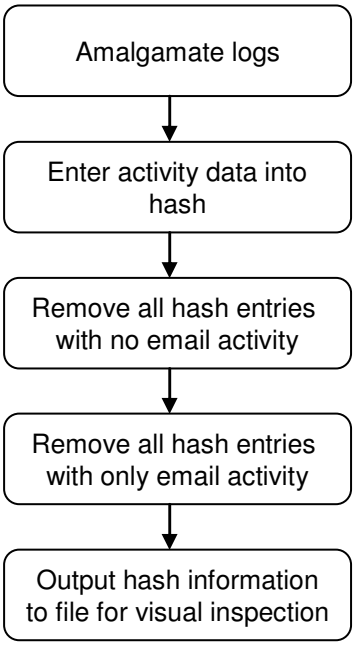

Figure 4.2: Phase 1 operation

The first stage in the Phase 1 implementation was to amalgamate log data from all gateways and mail servers into a single file. Entries were ordered by date and time, simulating the output that would result if all gateways and mail servers wrote audit log data to the same file. This allows the system to analyse traffic occurring across

multiple gateways simultaneously, potentially identifying attacks that would be missed if only activity from a single network gateway was analysed.

The second stage was to traverse the audit log, analyse each line to determine if it contained relevant data and if so, then place this data into a *Perl* hash table data structure (essentially an associative array) indexed by IP address. Within each IP hash table entry, all log lines relevant to that IP address were stored in an array. A hash table was used as it offered increased searching speed in data entry and extraction stages at the expense of high memory utilisation. Memory utilisation was not of high concern as this Phase 1 system was never intended to run in real-time; it was more important to ensure that the system could operate as fast as possible in order to generate results in good time.

Once activity data for all IP addresses was contained within the hash table, the next stage of the system involved reducing the amount of data in the hash table to only include data relevant to this investigation. The first step in the data reduction stage was to remove all IP entries from the hash table that contained no email activity. As the system that the audit logs supplied for this research used the Sendmail Mail Transfer Agent, this involved checking IP address entries in the hash for lines of the type *sendmail*. No email activity meant that the IP has not sent any spam (in the time period covered by the log file) and was therefore not relevant to our results. The second step in the data reduction stage was to remove all IP entries that only contained email activity. The logic behind this was that we are only interested in IP addresses that have other precursor network activity entries.

The final stage of the Phase 1 system was to print the remaining hash entries into an output text document for hand sorting and visual inspection. This process involved manually sorting through the result set as well as the use of UNIX commands such as *grep, wc* and *awk* to collect statistical information. Grep searches files for lines containing a match to a given pattern while awk provides further pattern matching functions. The UNIX word count (wc) function was also used, mainly to count the number of lines produced by the output of grep and awk statements. This allowed an insight to be gained into the frequency and types of network precursors to spam that appear in audit logs.

4.3 Phase 2 – Real-Time Operation

The Phase 1 system was able to analyse audit logs from multiple gateways and mail servers to determine if and what precursors appeared before spam messages arrived. This system did not operate in real-time, however, as it had to index an entire log file into the hash table before potential precursor activity could be identified through manual inspection of the program's output. The goal for Phase 2 was to construct a system that can operate in real-time, dynamically detecting new spam precursors and taking appropriate action against IP addresses that are identified as sending spam. Once a precursor pattern was determined, the system needed to detect the occurrence of precursor activity and block all email from the offending IP address from entering the network. This provides the system with the flexibility to adapt to the ever changing landscape of network precursors to spam.

The Phase 2 system was built upon techniques developed for the Phase 1 system, making the necessary modifications for it to run in real-time. One problem with Phase 1 which prevented fully automated operation was the identification of spam. As noted earlier, visual inspection of email headers was used in the Phase 1 system. For a real-time system, a method is needed to allow automated determination of spam messages. The accuracy of this detection method is also important as the accuracy of the entire system depends in part upon this initial spam detection process. The obvious choice for spam detection at this stage is the mail server's own spam filter. The use of this identification technique results in a much greater degree of certainty that a particular email is spam than could have been gained through the limited information contained in the mail server log. Also, as the results obtained from Phase 1 testing indicated precursors were only observed in a very small proportion of emails (Section 5.1), it was decided that a new method for detecting the impending arrival of spam was needed (Section 5.2.1). It was proposed that spam messages themselves could be used as precursors to spam, potentially indicating that the IP address that sent the original message was likely to send more.

Another necessary modification of the Phase 1 system was to move away from the large hash to store log information. The hash table created in the Phase 1 system consumed far too much memory to be used on operating gateways. It was decided

that a better solution for a real-time system was to store this information in a database. While the time required to insert and extract data from a database is significantly greater than using a hash table set up in memory, the use of a database offers greater robustness and flexibility. This robustness is provided by the fact that a database stores information on secondary storage, whereas the hash was contained solely in memory. If the computer running the program crashed or had to be rebooted for another reason, all the data stored in the hash table would be lost and the program would have to be restarted with no knowledge of previous events. Conversely, the use of a database would allow the program to start from where it left off after the computer was restarted as all of the data stored would still exist on secondary storage.

The use of a database also allows the system to be more flexible, as many different processes can access the data stored within the database. This provides support for separate programs to take action based on data contained within the database and also perform cleaning functions to prevent the database tables and firewall rules from using too many resources.

The last major modification required of the Phase 1 system involved taking action against those IP addresses that are sending spam. As the Phase 1 system was intended to identify the existence and attributes of spam precursors, no consideration was made in regards to what the system will do once a precursor has been identified. It must be noted that mail servers, especially those of large ISPs could very possibly be sending a mixture of both spam and legitimate email. A mechanism was needed to ensure that action is only taken against IP addresses that send a large proportion of spam or those that do not send legitimate email. To this end, the technique of Sequential Hypothesis Testing (Wald 1947) was used in a similar way to Jung et al. (2004) in the development of their Threshold Random Walk algorithm. Essentially, Sequential Hypothesis Testing is a method for defining two hypotheses (a simple hypothesis and an alternative) and testing via successive observations to determine whether either hypothesis has been reached. In the case of the Phase 2 system, a selection between two hypotheses was made, namely that a given IP address is either malicious or benign. This was accomplished by calculating and assigning suspicion values to IP addresses. Only once an IP address has been classified as malicious (i.e.

exceeded a particular spam suspicion threshold) will action be taken to block traffic from that IP into the network. Also, if an IP address is classified as benign, then the system will not monitor any messages from this IP for a period of time, effectively letting them all go on to the mail server. This provides the system with the scope to allow for large ISPs with a small amount of spammers to not have all of their legitimate email blocked and should at least minimise or perhaps eliminate the possibility that legitimate emails will be blocked by the system. This is an important issue as the blocking of legitimate emails affects the public's willingness to use the technology, as mentioned in Section 1.2.

4.3.1 Phase 2 System Overview

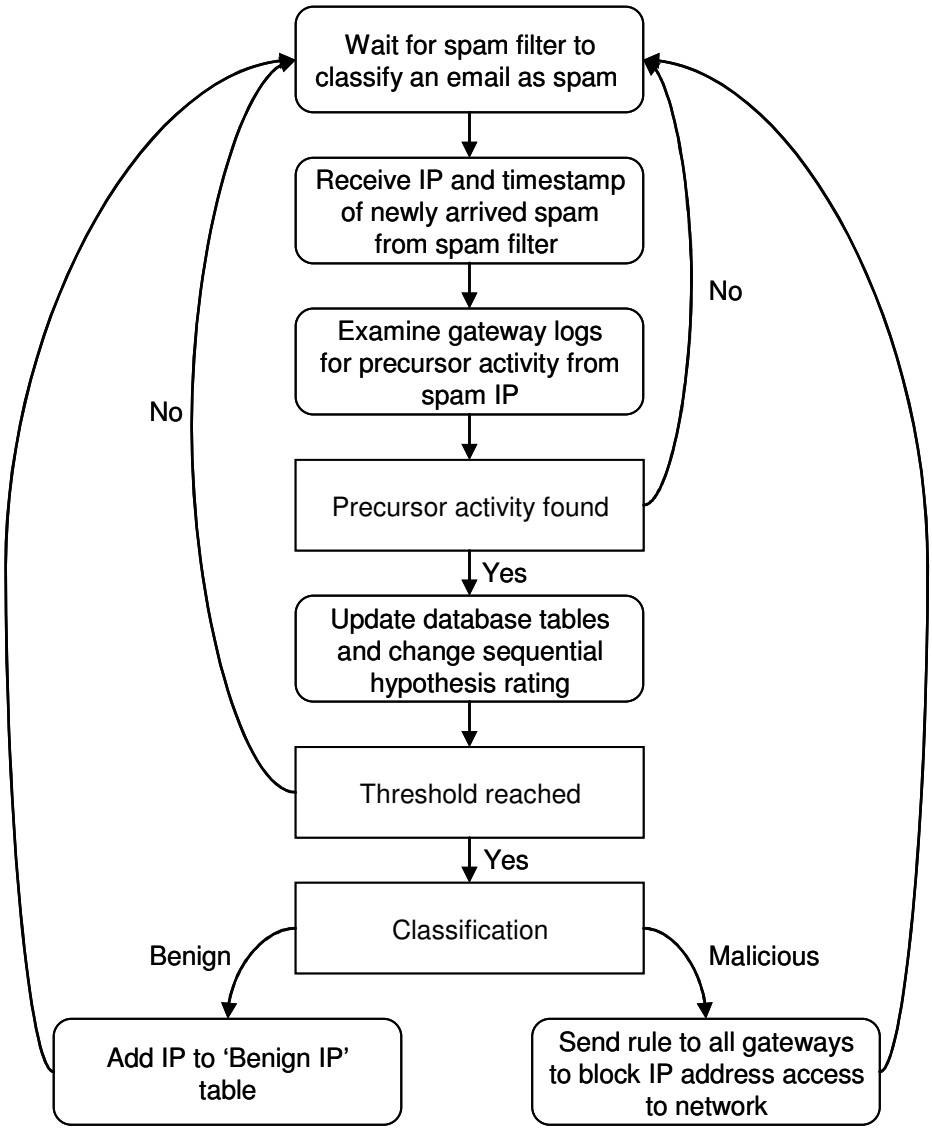

Figure 4.3: Phase 2 system operation

System Operation and Database Structure

In its default state, the system has to wait for the spam filter to classify a received email as spam. This is accomplished through the use of *Perl*'s *Tail* module (Grabnar 2004). Tail allows the system to monitor the spam filter log file and process each entry as it is added to the log. When a new log line appears, the system uses a Perl regular expression to extract the timestamp and IP address related to the spam message. Using Tail allows the system to approximate real-time operation, as it simulates monitoring an audit log file that is constantly changing. A script was created to dynamically add lines from a static source log to this file, simulating a spam filter writing entries to a log file. It was decided that there would be no limit placed on the speed at which the simulated log file was filled as this represents the maximum computational throughput the system would be subjected to. This also helped to expose any previously unseen inefficiency in the system.

Once a newly-added line from the simulated spam filter log has been processed and the timestamp and IP address extracted, the system then searches the gateway log up until the time of the spam filter log entry and adds all network activity from the spam IP address into the database. The amount of time to look back in the log files to search for precursor activity is a major determinant in the efficiency and accuracy of the system. Due to this, a major focus of this thesis is determining a log look back time that will offer the best compromise between system efficiency and accuracy. For this step, two database tables are used. One table (*IP Activity*) stores information relating to each unique IP flagged by the spam filter. This information includes the timestamps of the first and latest activity recorded from this address, the amount of portscans received from this address, the number of emails received from this address as well as the byte locations in the audit log of first and latest activity for the particular IP. This table also includes the address's current suspicion classification, which is used to classify IPs as malicious or benign.

The second table used for the storage of data in this step is the *Precursor* table. This table holds data relating to detected precursors, including the port numbers probed by each IP address and the number of times each port has been probed. This table is not currently essential to the operation of the system but it allows tracking of different

types of precursor attacks which may be valuable for future research. If the system already has precursor activity recorded for the IP address, the search through the gateway log will start at the line after the line relating to the last recorded precursor activity. This prevents the system from recording the same precursor activity multiple times per unique IP. This also increases the efficiency of operation of the system, as it does not have to process the same piece of data for a single IP multiple times. The structure of the *IP Activity* table is shown in Table 4.1 below.

IP1-IP4	These four fields hold the four octets of an IP address that the system is tracking. Splitting an IP address up into octet sections also allows for searching based on the whole IP and also different domain segments. The four IP address fields make up the primary key for this table.
First Activity	This field stores the timestamp of the first activity recorded for this IP. This, in combination with the Last Activity field allows the tracking of the activity lifespan of an IP
Last Activity	This field stores the timestamp of the latest activity recorded for this IP.
Mail Count	The count of emails (spam and non-spam) received from this IP
Scan Count	The count of port scans (network precursor activity) received from this IP
Suspicion	This field stores the current suspicion value of this IP. Once this value passes a certain threshold (determined by Wald's Sequential Hypothesis Testing), it is added to the *Benign IP* or *Banned IP* table.
First Log Byte	This field stores the byte location of the first activity recorded for this IP
Last Log Byte	This field stores the byte location of the last activity recorded for this IP.

Table 4.1: IP Activity table description

IP Suspicion

For every instance of network activity observed for a particular IP address, the suspicion value of the IP is changed. Malicious activity (such as a portscan or an

incoming spam message) increases an IP's suspicion level, while non-malicious activity (such as legitimate email) decreases it. If the IP address exceeds either of two predefined threshold values, it is classified malicious or benign, depending on which threshold was exceeded. If the IP address has been classified as benign, it is added to the *Benign IP* table in the database. This table records IP addresses that have been recognised as providing sufficient evidence that they are not malicious, allowing their mail to access the network without further interference by the system. If an email is received from an IP address listed in the *Benign IP* table, it is let through and not scrutinised any further. Precursor activity is not recorded for IP addresses in the *Benign IP* table. Once the IP is removed from the table, it is again subject to scrutiny by the system. The inclusion of the *Benign IP* table allows the system to operate with more efficiency, as IP addresses that send large amounts of legitimate mail and exhibit no malicious activity or only a minimal amount will be quickly added to the table, allowing the system to focus on more threatening IP addresses.

Taking Action

In a normal network environment, once an IP address is classified as malicious, the system would then communicate this information to all gateways in the network. Rules would then be added to each gateway's firewall to deny traffic from the malicious address entry to the network. As the system developed for this thesis acts in a simulated environment, the actual adding of rules to network gateways can not occur. Instead, this step is simulated by adding data to the *Banned IP* table. The system checks the IP address of incoming network activity and if the same address is listed in the Banned table then that packet would have been blocked from entering the network. For research purposes, the Phase 2 system records details of these packets to another database table, the *Blocked* table. This simulates the blocking of messages and also allows empirical data to be gathered about the amount of emails blocked by the system and the IP addresses they came from. It must be noted that in a real world system these packets would have never even entered the network. Table 4.2 (below) shows a description of the structure of the *Banned IP* and *Benign IP* tables.

IP1-IP4	These four fields hold the four octets of an IP address that the system is banned or rated as benign. Splitting an IP address up into octet sections also allows for searching based on the whole IP and also different domain segments. The four IP address fields make up the primary key for this table.
Start	This field records the time that this particular IP address was added to the *Banned IP* or *Benign IP* tables
End	This field records the time that this particular IP address is due to be removed from this table.
Suspicion	This field records the IP's suspicion value at the time of being added to the *Banned IP* or *Benign IP* table.

Table 4.2: Banned IP/Benign IP table description

As the system is intended to run in real-time, ensuring efficiency is essential. Every extra database entry adds a small amount of seek time per database query. Also, every firewall rule added at the gateways increases the amount of time needed to filter every packet attempting to access the network. Therefore, the key to keeping the system running efficiently is to remove database entries and firewall rules that are no longer needed. In the Phase 2 system, this was accomplished through the use of limited ban lengths and the *History* table. When an IP address exceeds one of the two suspicion thresholds and is classified either benign or malicious, a variable in the system determines how long a given IP address will be banned from the network or classified as benign. Once this time limit has been reached, the IP has its network ban lifted or its benign status revoked. In reality, the removal of a ban would involve sending messages to all gateways in the network to remove the rule relating to packets from the previously banned address. To simulate this, the Phase 2 system removes the entry from either the *Benign IP* or *Banned IP* tables and adds it to the *History* table. The *History* table keeps track of IP addresses that have been either banned from the network or rated as benign. This allows statistics to be collected about the amount of times IP addresses have been banned from the network or rated as benign.

Setting a limit on the amount of time an address will be rated malicious or benign also deals with the possibility of a benign IP becoming more malicious over time or a malicious IP reforming and becoming benign. Considering this possibility is very important, as something as simple as an ISP changing its email policies (e.g. closing open relays) could turn a domain known for sending spam into a domain from which little spam comes. This gives the system the flexibility to dynamically determine if an IP is malicious or benign, as opposed to systems such as blacklists that produce a single static verdict. Table 4.3 details the structure of the *History* table.

IP1-IP4	These four fields hold the four octets of an IP address that have previously been banned from the system or rated benign. Splitting an IP address up into octet sections also allows for searching based on the whole IP and also different domain segments. The four IP address fields make up the primary key for this table.
Banned Count	This field stores a count of the number of times this IP has been banned from the network.
Benign Count	This field stores a count of the number of times this IP has been rated as benign by the system.
Total Suspicion	This field stores the total suspicion value of the IP address.
Last Activity	This field stores the last time a given IP entry was added to the table.

Table 4.3: History table description

To ensure correctness of the both the Phase 1 and Phase 2 systems, they were tested with synthesized logs made up of known quantities of legitimate email, spam email and precursor activity. The results from these tests were hand checked to ensure proper output was generated. The source code to the Phase 1 and Phase 2 systems, as well as the scripts to create the aforementioned database tables can be found on the Appendix CD accompanying this thesis.

4.4 Testing

Testing for both the Phase 1 and Phase 2 system was carried out on live audit log data. The audit logs that have been provided for this research project come from a network covering an almost complete class C IP domain. This network is similar in size to a typical small hosting provider or business in Australia. Internally, the network is comprised of multiple gateways connected to multiple mail servers.

Four audit logs from the above system were used as the primary sources of data for this research. These audit logs contain syslog entries amalgamated with entries from the mail servers. Each of these audit log files are comprised of contributions from the Sendmail mail transfer agent, the PortSentry intrusion detection system and the Linux kernel firewall (running iptables[4]). Details about the types of information reported by these sources can be found in Section 2.6.1 and Section 2.6.2. The first audit log file encompasses a 21 day period from the 1st to the 21st of July 2004. The second audit log covers a 31 day period between the 1st and 31st of August 2004. The third log used in this research covers a 26 day period between the 1st and 26th of July 2005 and the forth log contains activity recorded between the 1st and 31st of August 2005. Table 4.4 shows the differences between the various audit log files used for this research.

	Month	Year	File Size	Length	Total Lines	Sendmail Lines
Log 1	July	2004	296 MB	21 Days	2203331	141663
Log 2	August	2004	210 MB	31 Days	1559537	110201
Log 3	July	2005	94 MB	26 Days	740761	189605
Log 4	August	2005	300 MB	31 Days	2198704	363260

Table 4.4: Audit log file statistics

4.4.1 Phase 1 Testing

The first sets of results represent a validation of MacLeavy's previous work and used Logs 1 and 2. Log 1 and Log 2 are the exact files that were used by MacLeavy when gathering his results. The major issue that was identified when recording results

[4] http://www.netfilter.org/projects/iptables/index.html

from Phase 1 testing was that of spam classification. The Sendmail entries in the log file contained only a small amount of information about each received mail message. It was decided that three fields of the Sendmail entry were to be used to determine spam; *from*, *msgid*, and *relay*. Based on the information listed in each of these fields, a decision was made as to whether a particular message was spam. If a message had no *from* address, for example, it most likely was spam, as legitimate email programs generally include this information when an email is sent. Also, the domain information included in the *from* address (if it contains any data at all) was checked against the domain listed in the *msgid* and *relay* fields; inconsistent domain information can be an indicator that an email is spam, although it must be noted that this is not always the case. In order to take this into account, *msgid* and *relay* information was only examined if the *from* address looked suspicious or contained no information.

4.4.2 Phase 2 Testing

The results recorded for the Phase 2 system were obtained from testing on newer audit log files, namely Logs 3 and 4. These log filed were chosen primarily because they show network activity from 2005, as opposed to Log 1 and Log 2. It was decided that the most recent logs would provide a better picture of current network activity than logs that are greater than one year old. It was originally intended that the system would be tested on live spam logs from the time period covered by Log 3 and Log 4. Unfortunately, due to the way the logging for the spam filter that provided logs for this research was configured, it was extremely difficult to extract the correct IP address of emails that the spam filter had flagged as spam. Instead, simulated spam filter logs were created from mail server logs. As it is widely recognised that emails sent with no information in the *from* field have a very high probability of being spam, emails with no information in the *from* field were deemed to be spam and added to the simulated spam log. In essence, this simulated spam log represents the most basic of all spam filters that could be installed on a mail server. While not ideal in a pure spam identification sense, this simulated log should still provide sufficient evidence as to whether the system is feasible.

The different values of the suspicion and threshold variables can be used by an administrator to represent a network's email policy. The relationship between the

spam and *other email* values represents the proportion of traffic required for a particular IP address to be added as benign or banned from the network. The thresholds can also be modified according to policy, as they represent the 'amount of evidence' the system requires in order to classify an IP. Set higher and the system can be more assured that it knows an IP's real intentions, at the cost of letting more spam through initially. Conversely, setting either threshold lower decreases the overall accuracy of the system in order to respond to a potential spam attack more quickly.

Spam Value	Other Email	Port Scan	Ban Threshold	Benign Threshold	Ban / Benign Length
+4	-2	+1	10	-10	24 hours

Table 4.5: System variables

The values used for Sequential Hypothesis testing in this research are given in Table 4.5. These variables all have an effect on system operation. The *spam*, *other email* and *port scan* values all determine how effective the system will be in both identifying spammer IP addresses that should be banned from the network and also identifying benign IP addresses that should be allowed access. The *spam* value was set as contributing +4 to an IP's suspicion, on the basis that an IP should be banned once it sends three spam messages, with up to a single legitimate mail accompanying these. The *port scan* value is set as contributing +1 to an IP's suspicion value on the basis that receiving ten precursors, one spam and six precursors or two spams and two precursors from a spammer IP should result in a system ban. Conversely, the *other email* value was set as contributing -2 to the IP address's suspicion value. This was set as a lower value mostly because the system classifies all emails that are not expressly flagged as spam by the spam filter as legitimate. Due to the fact that this system had to be tested on synthetic spam logs representing the most rudimentary spam filter, it is extremely likely that many other emails shown in the logs are also spam. The lower value for the *other mail* variable is an attempt to offset the issue that there would be a much higher amount of false negatives produced by the system as a result of the inaccurate spam identification method used. This way, an IP address needs to send the system at least five emails that are not flagged as spam before being counted as benign. A system with a more accurate spam detection

method would benefit from having this value set closer to the negative of the spam email value, in this case -4 might prove to be a better value.

The length of time that an IP is classified benign or banned from the network has been set at a static 24 hour time period. This was used mainly because it is a common default set by system administrators who use intrusion detection systems (Manderson 2005). Also, this value sets a safeguard for the potential banning of legitimate email sources as the maximum amount of time an IP address can be banned from the network is this 24 hour time period.

The Phase 2 system was run multiple times to extract results for this research. The first run for each log file had no restrictions placed on how far it looked back in the audit log. Also, the system variables were set such that IP addresses could never be classified as malicious or benign. These settings allowed statistics to be collected from the system running at maximum accuracy, and provided a baseline processing time that other system modifications could be compared against. The system was then run with IP classification activated at various log look back lengths, from as long as a full month to as little as a single day. These tests provided the data that is presented and analysed in the Results and Discussion section.

5 Results and Discussion

The following section examines the results obtained from tests performed on both the Phase 1 and Phase 2 systems. The audit log files used to perform the tests have been provided by Kevin Manderson from his Security Consulting Business. The details of the Phase 1 and 2 systems, as well as the audit logs used can be found in the Methodology chapter of this thesis (Section 4).

5.1 Phase 1 Results

In contrast to the eight days of server processing needed to extract results from the MacLeavy system, the revised Phase 1 system took just nine minutes and eight seconds to process the entirety of Log 1 and only five minutes and five seconds to process the entirety of Log 2. This significant increase in run time efficiency can be attributed to a variety of factors. The first and most significant factor was that the searching algorithm used was more efficient. While the MacLeavy system had to iterate through the audit log file once for each mail encountered, the Phase 1 system only required a single pass through the log file in order to store activity data into the hash table. This did result in higher memory utilisation, but still shows that real-time operation of a similar system is more feasible than the MacLeavy results suggest. Also, it must be noted that the results recorded for the Phase 1 system were obtained using an 800 MHz Pentium 3 with 512 MB of memory, a decidedly inferior system to the server that the MacLeavy system ran on. This further highlights the Phase 1 system's increased efficiency.

Table 5.1 (over page) shows some statistics that were collected in the course of examining the MacLeavy results. In Log 1, 123 email messages from 98 distinct IP addresses were recorded as having precursor activity. An average of 6.98 precursor scans were received per message that exhibited precursor activity. Additionally, the results recorded for Log 2 show 223 distinct IP addresses sending a total of 312 email messages with precursors, with the average precursor scans per received email as 10.29. This clearly shows that precursor activity for email messages was

observed, although the proportion of messages recorded with precursor activity was extremely small when compared to the overall count of mail in the time period covered by the audit log. Only 0.09% of emails from Log 1 contained any network precursor activity at all and Log 2, although offering a significant precursor increase over Log 1, still only reports precursors for 0.28% of all emails. It needs to be noted that these figures represent statistics for precursors for all email messages, not just spam. As Logs 1 and 2 contained approximately 80% spam compared to regular email (Manderson 2005), the proportions presented by this initial investigation were deemed sufficient enough for the determination to be made that port scan precursor detection would not be useful to the real-time (Phase 2) system.

	Log 1	Log 2
Total mail count	141663	110201
Mail with precursors	123	312
Precursor count	858	3211
Average number of precursors for mail with precursors	6.98	10.29
Average number of precursors for all mail	0.006	0.029
Distinct IPs with precursor activity	98	223

Table 5.1: Phase 1 precursor statistics

The major ports that were scanned as precursors to the sending of mail, as shown in Table 5.2 (over page), were 135, 139, 1433, 25 and 0. TCP port 135 is officially used for Microsoft Remote Procedure Call (also known as Distributed COM Service Control Manager), while port 139 is officially used for the NetBIOS Session Service (Seifried 2003). These services are both used for SMB file and print sharing, but more recently TCP ports 153 and 139 have received attention as ports exploited by worms specifically targeting Microsoft Windows users, such as the W32/Blaster worm (Dougherty et al. 2003). Port 1433 is reserved for use by the Microsoft SQL server but is also targeted for exploitation by another Internet worm known as Spida, SQLSnake or Digispid (Dougherty and Householder 2002). The release of these worms predates the period of time covered by Logs 1 and 2 and thus could be responsible for scan activity recorded over this time period.

	Number of scans per distinct port	
Port Number	Log 1	Log 2
0	82	56
25	157	0
135	398	263
139	215	2845
1433	6	47

Table 5.2: Phase 1 precursor ports

Other precursors appear on TCP ports 25 and 0. Port 25 is used for SMTP connections (i.e. the transfer of email messages) so activity on this port could represent a remote entity scanning to see if a mail server is operating. Port 0 is listed as a reserved port by the Internet Assigned Numbers Authority[5], meaning that no activity should appear on this port. Even though it is reserved, some systems treat a port 0 connection attempt as a request for connection on the lowest free TCP port. Additionally, due to the differing responses generated by various operating systems when a port 0 connection is requested, port 0 scanning can also be used to determine the OS running on a target machine (Jones 2003). This observed activity on port 0 could represent an automated mailer testing to see if a computer is located at the target IP address, then sending spam to that address if the target is confirmed.

Another observed pattern was that for mail that appeared after precursor attacks consistent with worm activity (ports 135, 139 and 1433), the message usually arrived a number of days after the last precursor activity. Conversely, mail that appeared after port 0 and 25 activity usually arrived within a number of seconds.

From the above results, it could be inferred that precursor activity on the commonly exploited ports is indicative of traffic arriving from zombie machines compromised by the worms listed earlier. The initial (precursor) activity from these hosts could represent the worm attempting to propagate itself on to other machines, while the subsequent spam messages result from the compromised machines being utilised to send spam. As the precursor activity observed on ports 0 and 25 occurred much closer to the actual arrival of spam messages, this activity could indicate the use of a bulk mailer program that scans for active IP addresses to propagate spam to.

[5] http://www.iana.org/

If the activity on ports 135, 159 and 1433 does indeed represent precursor activity from a computer compromised by a worm, audit log based precursor detection could be a very useful way in which to detect the IP addresses of zombie machines. A list of these zombie computers could be maintained and network gateways could block all activity from these machines for a period of time, say 24 hours. These lists could also possibly be made public so other network administrators could also be informed of potential zombie addresses. This could be an interesting avenue for future investigation.

Despite the findings above, the fact that precursor activity was only observed in a small number of all emails indicates that detecting actual spam based on precursor activity alone will most likely offer only a minute improvement on current systems, at a significant increase in computing power needed to run the system. For the real-time system to be viable, an alternate method of determining from whom spam is likely to arrive needs to be developed.

5.2 Phase 2 Results

The Phase 2 system is designed to test whether precursor detection at the gateway is a viable spam protection technique. To answer this, two areas need to be examined: timeliness and effectiveness. It needs to be determined if the Phase 2 system can operate fast enough to be viable as a real-time system. It also needs to be determined if the system will block enough spam messages to warrant the drain it places on gateway resources. If the system only manages to block a small proportion of spam messages, it would make more sense to let the mail server filter every message (as it currently does) and not increase the load on the gateways unnecessarily. The amount of time that the system looks back in the amalgamated audit log influences the timeliness of the system, as more processing is required every time the system searches for precursor activity for a particular IP address. The effectiveness of the system is also influenced by the amount of time used in precursor searching, as the further back an audit log is searched, the more accurate the detection will be. Therefore, it is important to determine a value representing the amount of time in which to search back through the audit log which offers a fast operating time combined with accurate precursor detection. If this amount of time is too little,

precursors could be missed while if the time is too high, the system may be late in informing other gateways to block incoming spam, lowering blocking effectiveness.

5.2.1 Spam as a Precursor

Due to the low percentage of network activity observed as spam precursors in the results from Phase 1 (see Section 5.1), an alternative method was sought to determine if spam is likely to be received by the network. The solution implemented in the Phase 2 system was to use spam messages themselves as an indicator that more spam is likely to arrive from an IP. This allows the system to identify IP addresses that are sending multiple spam messages to different addresses within the network or sending multiple spam messages to the same address. A criticism of this approach could be that it requires the network to receive multiple spam messages before action is taken. While this is true and the obvious solution could be to just ban a spammer's IP address as soon as the first spam is flagged by the mail server's spam filter, it is believed that the method implemented provides the end user with greater protection from false positives. As spam filtering at the network wide level generally involves no user input at all, it is important to do as much as possible to prevent legitimate email from being marked as spam and blocked from the network.

Concentrating on spam as precursor activity as well as monitoring other network precursors proved appropriate, as only three network precursors were recorded out of 6977 spam messages analysed from Log 3. Log 4 only fared marginally better, with twelve precursors recorded out of the 4420 spam messages analysed. With numbers this low, it cannot be conclusively stated that this activity represents specific precursor activity to the sending of spam.

5.2.2 Time Span of IP Activity

In order to determine how far to look back in the system audit logs, the amount of time that IP addresses are usually active needs to be examined. Complete activity times were calculated for every unique IP that sent email. This was found by calculating the difference between the time of latest activity and the time of first activity for each IP. These times were then grouped together in discreet day periods, from less than one day to between 15 and 31 days. For example, IPs with an activity time span of less than a single day were grouped together and IPs with an activity

time span of between one and three days were grouped together. This process was completed until each IP was contained within one of these groups. The pie charts in Figure 5.1 and Figure 5.2 below show the relative proportions of IP activity time over the two log files, calculated from information contained in the *firstactivity* and *lastactivity* fields of the *IP Activity* database table.

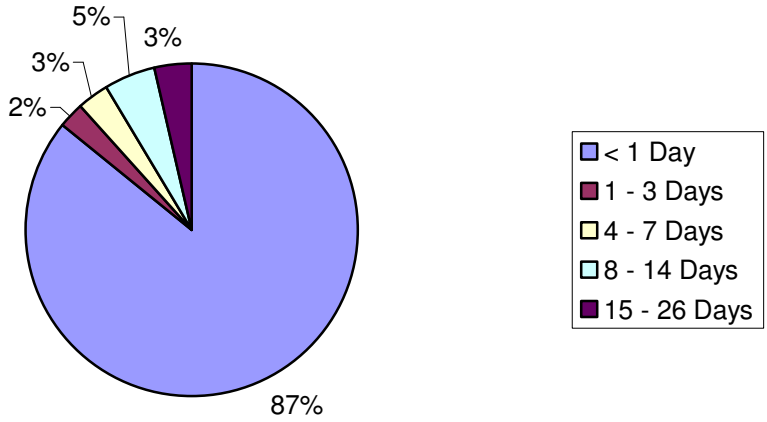

Figure 5.1: Log 3 IP activity time span (July-05)

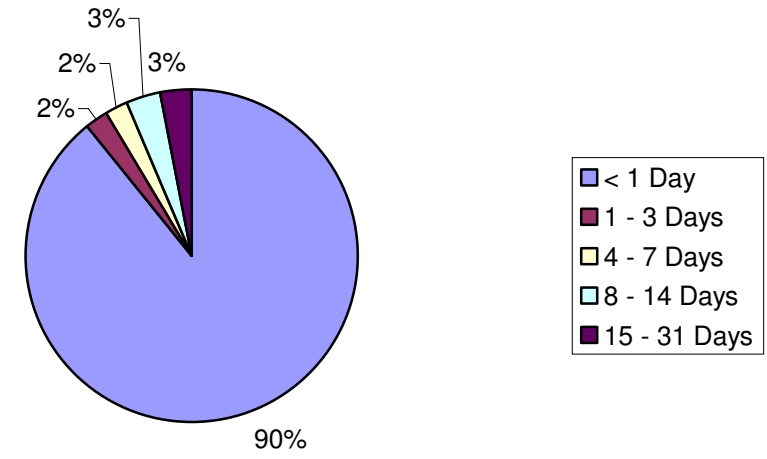

Figure 5.2: Log 4 IP activity time span (Aug-05)

What is immediately apparent is that an overwhelming majority of the IP addresses recorded had a total activity time span of less than a single day. Furthermore, over 90% of IPs observed have an activity time span of less than a week. Figure 5.3

shows the proportion of IPs that would have their complete activity profile observed for each amount of days looked back.

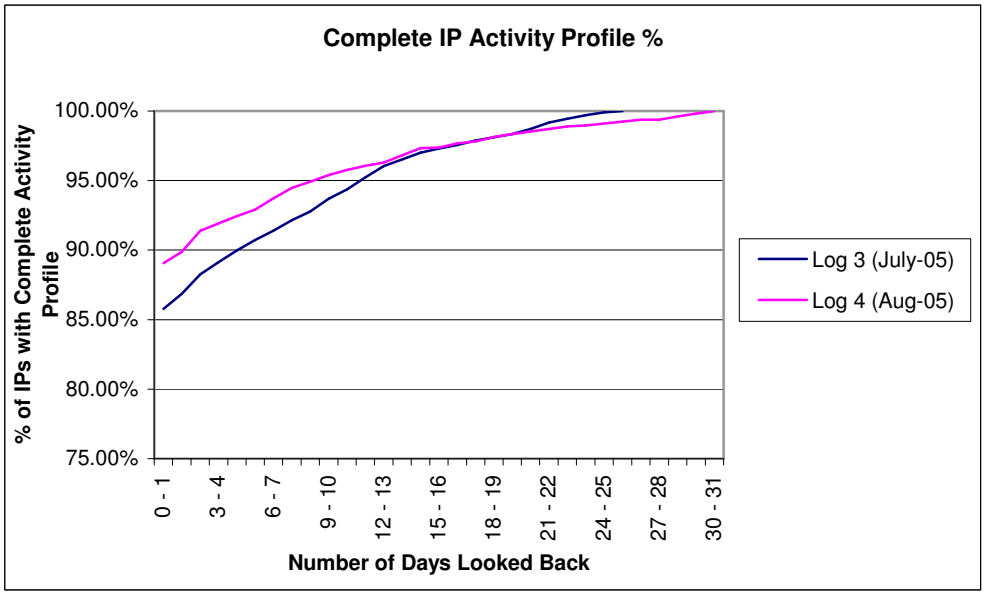

Figure 5.3: Amount of IP activity observed for different look back amounts

As can be seen in the above graph, looking back 5 days means the system will observe the full activity time spans of over 90% of all active IP addresses in the log file, while looking back 12 days allows the system to fully observe over 95% of all active IPs. Furthermore, looking back more than 12 to 14 days yields only minimal percentage increases for each extra look-back day. This suggests that the amount of time to look back should definitely be set at a period of less than two weeks, as looking back any extra amount of time makes the system increasingly inefficient for only small increases in accuracy.

5.2.3 Time Cost of Looking Back

Another factor that plays a part in the determination of how far should be looked back in the logs is the amount of time it takes for the system to process each new spam message. If this time is too large, the system will not be able to act in time to stop the arrival of subsequent spam. For every entry in the simulated spam log, the time the system took to process this entry was recorded. The results for this test on Log 4 (the more complete of the two logs used for Phase 2 testing) are showed in Figure 5.4 (over page).

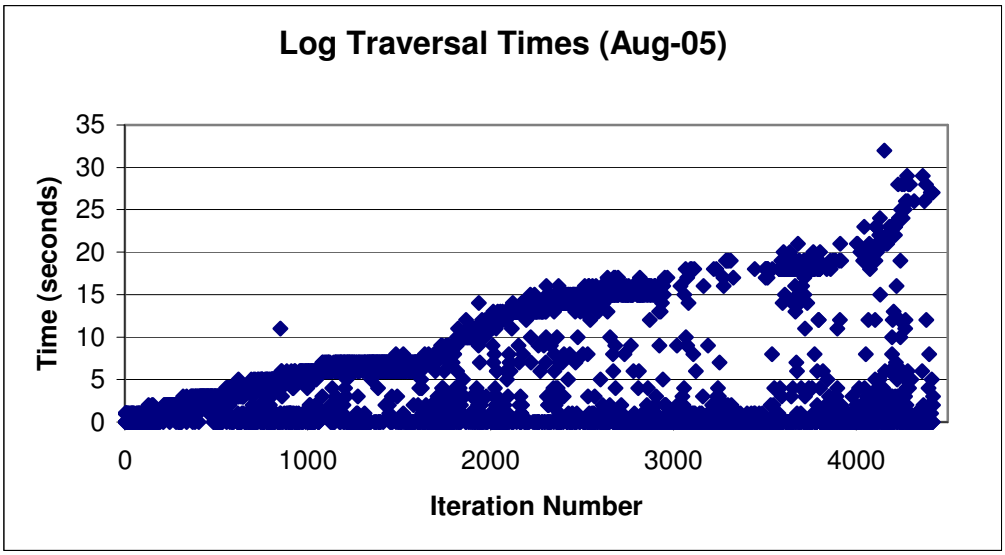

Figure 5.4: Look back times for spam messages in Log 4 (Aug-05)

In this test, the amount of time the system looks back in the log file was unrestricted. As the above iteration times were collected from a static single month log file, the amount of log look back time is dependant on the time of activity. For example, a spam message that arrived on the 2nd of August only had two days for the system to look back, whereas a spam message that arrived on the 28th of August still required the system to search from the start of the log and therefore had to search for 28 log days. As the iterations were ordered by date and time of arrival, the earlier iterations represent activity from earlier in August whereas the latter iterations represent activity from later in the month.

It can be seen that the amount of time that the system takes to complete an iteration rises in a generally linear fashion as the amount of time that is looked back in the log increases. The plateau sections, particularly between iterations 1000 and 1500 are caused by a greater density of spam activity condensed within a particular time period, whereas the sections with a greater than normal gradient (for example, between iterations 4000 and 4500) represent a period of time with less spam activity than normal. The iterations shown on the graph that take less than the normal amount of time represent extra efficiency built in to the system. This efficiency is mostly due to the fact that the system does not have to start at the beginning of the log file when it encounters an IP address that it has seen before (see Section 4.3.1).

The large proportion of iterations that took less time to complete indicates that the system is operating at a high level of efficiency.

The occasional iterations with a time larger than normal (one at approximately iteration 900, one at approximately iteration 1950 and one at approximately iteration 4200) represent times when the system encountered a new benign IP. As the activity for an IP address is not examined until a spam message from that IP is received,
The first time a spam message is received from an IP with a large amount of legitimate email activity causes the system to examine all the activity for that IP, resulting in a large amount of time to process that iteration. These iterations represent the only times where system timeliness is not defined by the amount of time it takes to look back in the audit log. These situations are extremely rare though, with only three recorded out of the 4420 entries in the simulated spam log. Also, this only happens once for each new benign IP encountered, as the IP will be added to the Benign IP table once the initial iteration completes.

5.2.4 Total System Run-Time

The final measure of the timeliness of the system is to see how long it takes to fully process an entire audit log file. Theoretically, as long as the system finishes processing an audit log in less time than the time period covered by the log file, it can be said to be running in real-time. This being the case, it is still important to lower the running time as much as possible in order to increase the scalability of the system. Figure 5.5 below shows the total running time of the system for various log look back lengths.

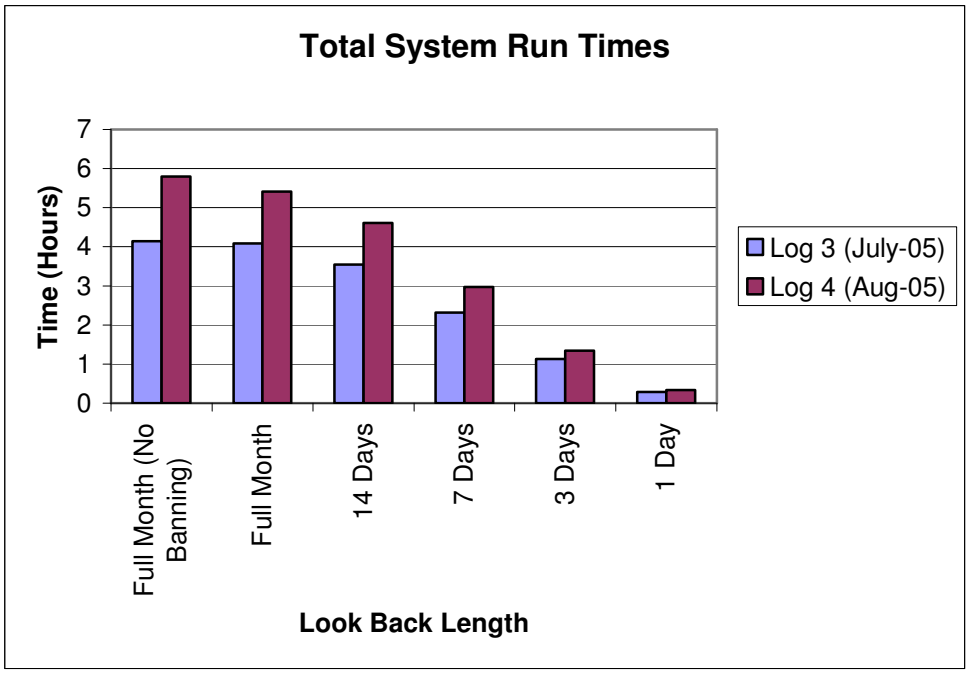

Figure 5.5: Total system run times

The first set of columns in the chart represents a system run with unlimited (full month) log look back and no banning (or benign classification) of IP addresses. This provides a base time for system operation that can be used when measuring the efficiency gains of banning of different log look back amounts. The next group represents the same look back amount, but with banning and benign monitoring activated. This lowered that actual amount of spam messages that the system had to process, but surprisingly did not lower the system run time significantly. This is most likely due to the way the simulated system still had to process messages from blocked IP addresses (see Section 4.3.1). It is expected that running the system in a real environment (as opposed to a simulated one) would show greater gains with IP banning activated. The difference in operation time between Log 3 and Log 4 is due to the fact that Log 3 covers a shorter time period (26 days) than Log 4 and that, due to various logging features being disabled by the administrator, has a lower activity density (see Section 4.4 for respective log file sizes). Despite this, both audit logs show a similar trend in the amount of efficiency gained by increasingly smaller log look back amounts.

Looking back 14 days instead of the whole month reduces the total running time by approximately 20%, while looking back a week only takes approximately 50% of the

run time of the full look back system. The smaller look back values further increased system efficiency, as the three day look back took only approximately 25% of the full run time while the single day look back system required only 20 minutes (5.86% of the original system run time) to process a full month of audit log activity. This shows that the system is definitely efficient enough to operate in real-time, with significant scope for handling larger networks with audit logs of greater density.

Using these time figures, it can be calculated that the single day look back system took on average 0.28 seconds to process a single spam message from Log 4. The three day system took 1.09 seconds to process a single spam message, while the seven day system took 2.42 seconds, the 14 day system took 3.76 seconds and the month long look back system took 4.41 seconds on average to process a single spam message.

5.2.5 System Effectiveness

Another factor to be examined in order to determine how far to look back in the audit log is the impact that this look back time has on system effectiveness. Figure 5.6 below shows how many IP addresses were classified malicious (and subsequently banned) or benign.

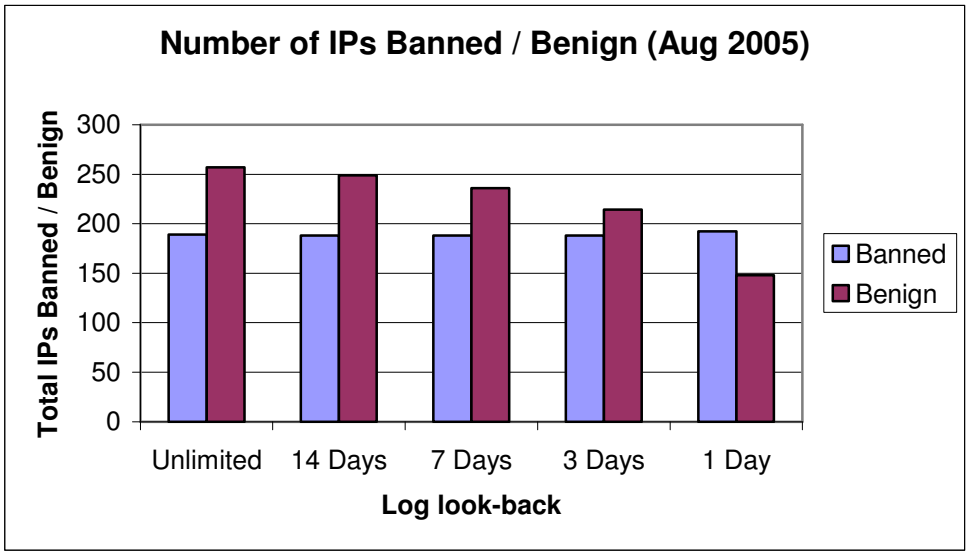

Figure 5.6: IPs banned / benign in Log 4 (Aug-05)

The graph above shows that the number of IP addresses classified as benign decreases as the number of days looked back in the audit log decreases. There is an

especially large decrease in IPs classified benign between the 3 day log look back and the single day log look back systems. This indicates that a significant proportion (approximately 42%) of IP addresses that have been rated as benign by the system have a total activity time span of greater than a single day, while 16.7% of IP addresses classified as benign have a total activity time span of greater than three days.

In comparison with the results of benign table additions, the number of IP addresses classified as malicious and subsequently banned from the network remain steady throughout all look back lengths tested, with the number actually rising slightly at a look back amount of one day. This indicates that almost all of the IP addresses that are classified as spam by the system have an activity time span of less than a day, as the system does not lose any banning accuracy by looking back shorter amounts. The slight rise in IP addresses banned for look back amounts of three and seven days is probably due a small amount of IPs that were classified as benign in the three day and above look back systems being now classified as malicious with only a single day of log look back. This shows that malicious IPs generally send all of their spam in a small period of time (less than a day) and then disappear from the network, whereas benign IPs keep in contact with the network for longer periods of time (as would be expected). This is essentially due to the one-way nature of spam; a spammer sends out a large volume of messages in the hope of some replies. Legitimate emails, on the other hand generally lead to replies back and forth, leading to a longer period of activity recorded for the legitimate sender's IP. Figure 5.7 (over page) shows the actual amount of email messages blocked (if the IP has been banned) or ignored (if the IP is benign) by the system for the same time period.

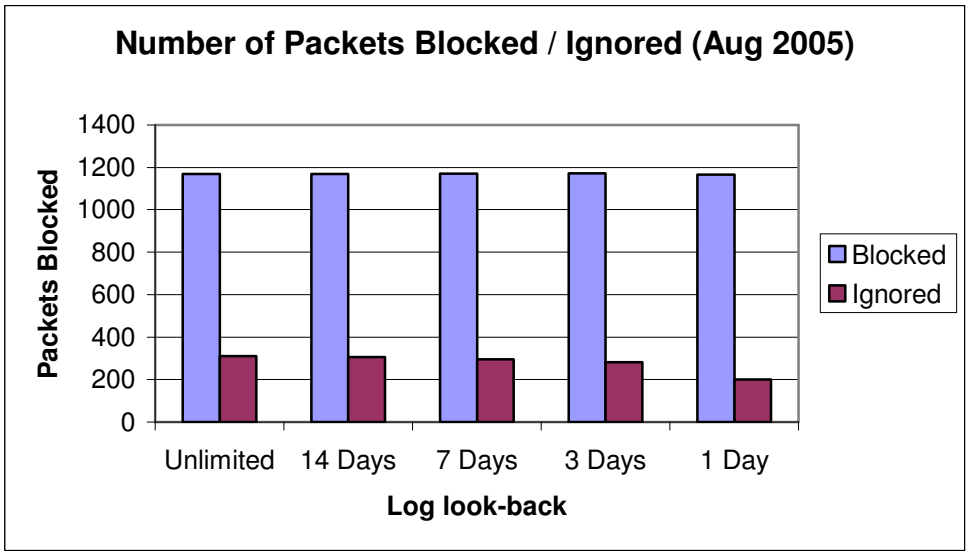

Figure 5.7: Number of emails blocked / ignored from Log 4 (Aug-05)

The above figure shows that the number of packets ignored by the system drops between the three day look back and one day look back systems. This further supports the conclusion that only looking back a single day in the audit logs leads to less accuracy in the detection of benign IPs. The amount of emails blocked by the system for different log look back times also does not change significantly, supporting the conclusion that the majority of spammers operate within a time period of less than 24 hours. The number of emails ignored by the system drops off between the three day and single day look back systems, partially supporting the information shown in Figure 5.6. Interestingly, the number of emails ignored does not reduce significantly between the unlimited look back run and the three day run. This suggests that a high percentage of benign IP addresses are active for a time period of between one and three days, as even though more IPs were classified as benign with a longer log look back, the amount of actual emails ignored only reduces slightly between an unlimited look back value and a three day look back value.

	Log Look Back Time				
	Unlimited	14 Days	7 Days	3 Days	1 Day
Total Simulated Spam	4420	4420	4420	4420	4420
Total Emails Blocked	1170	1170	1171	1172	1166
Block %	26.47%	26.47%	26.49%	26.52%	26.38%
Difference From Unlimited	NA	0	1	2	4
% Difference From Unlimited	NA	0.00%	0.09%	0.17%	0.34%
Number of Distinct Bans	189	188	188	188	192
Total Emails Ignored	310	306	295	282	200
Ignore Percentage	7.01%	6.92%	6.67%	6.38%	4.52%
Difference From Unlimited	NA	4	15	28	110
% Difference From Unlimited	NA	1.29%	4.84%	9.03%	35.48%
Distinct Benign Classifications	257	236	249	214	148

Table 5.3: Log 4 (Aug-05) blocked / ignored statistics

Table 5.3 shows the proportion of spam messages that were blocked by the system with the variables set as defined in Table 4.5. It can be seen that the system, if running on a real network, would have blocked over 25% of the spam from the simulated spam log from entering the network at all. This could represent a significant increase in the running efficiency of the network's mail servers as well as a reduction in overall internal network traffic. Overall, the unlimited look-back system blocked 1170 spam messages from 189 distinct bans, representing an average of 16 spam messages received per ban. It must be remembered that these percentages come from analysis of a simulated spam log, with all emails not flagged as spam by this simulated log classified as legitimate. Any real-world spam detection technology would detect more spam than the amount contained within the simulated log and the system should then increase in detection (and therefore blocking) accuracy. Another interesting observation that can be made from this table is that 35.48% less benign emails are ignored by the system with a single day log look back; while only 9.03% of benign emails are missed in the three day look back system. This further reinforces the conclusion that a significant proportion of benign activity occurs in a time period of between one and three days, thought the highest proportion of benign activity (64.52%) still occurs within a single day.

5.2.6 Time Between IP Ban and Subsequent Spam

A major factor that determines the viability of the system is whether it can detect and respond to an incoming spam attack in time to stop subsequent spam from the IP entering the network. To test this, 10% of IP addresses were randomly selected from the list of IPs banned from the network and subjected to further investigation. The amount of time between when each IP was banned and when the next email arrived from that IP was recorded, as well as the time details for the subsequent two emails for the same IP. Figure 5.8 shows the distribution of arrival times for the first message received after an IP has been banned, while Figure 5.9 shows the distribution of arrival times for the second and third messages received by the same IPs after they have been banned.

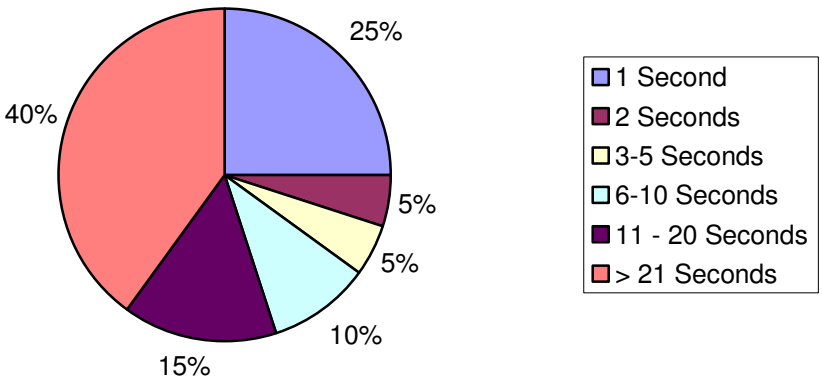

Figure 5.8: Arrival times for first spam after an IP is banned

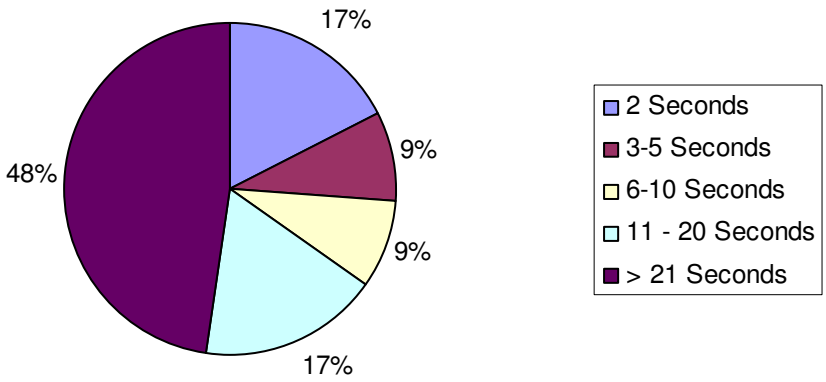

Figure 5.9: Arrival times for the second and third spam messages

Figure 5.8 shows that 25% of the IPs investigated sent spam within one second of being banned from the network and with a further 10% sent subsequent spam within five seconds of being banned. This shows that if the system can classify an IP as a spammer and block it from the network within five seconds (as this is greater than the average time it took a full month look back system to process a single spam message), it is be able to block the approximately 65% of spammer IPs completely, while 35% will be able to propagate at least one more spam message to the network before it is blocked. Furthermore, only 26% of the second and third messages send by banned spammers arrives at the network within five seconds. As each spammer sends an average of 16 mails per ban (as mentioned above), the amount of spam let through by the system represents only a small proportion of the amount of spam it would catch.

If the system can classify an IP as a spammer and block it from the network within two seconds, only 30% of IPs would be able to get a subsequent spam message through before being blocked. Also, only 17% of spammers would be able to propagate a second or third message before being blocked. As the three day look back system took just over a second on average (see Section 5.2.4) to process an incoming spam, it seems that this is a reasonable compromise between accuracy of spam determination and effectiveness of the system.

5.2.7 Other Findings

Repeat Offenders

Another factor that can influence the effectiveness of the system is the amount of times an IP is banned from the network or classified as benign. Table 5.4 and Table 5.5 show how many times IPs were added to the banned and benign tables.

Number of times banned	Distinct IPs
1	122
2	18
3	3
4	2
5	1
6	0
7	0
8	0
9	1

Table 5.4: Log 4 (Aug-05) ban frequency

Number of times benign	Distinct IPs
1	29
2	6
3	4
4	4
5	3
6	0
7	0
8	1
9	0
10	1
11	0
12	2
13	2
14	3
15	3
16	1

Table 5.5: Log 4 (Aug-05) benign frequency

It can be seen that the majority (82.99%) of IP address that are banned from the network are only banned once, suggesting that a large amount of spammers are attacking the network only a single time. Benign IPs are much more likely to be classified as benign more than once, with just over half (50.85%) of the IPs classified

by the system as benign added to the benign table multiple times. This suggests that IPs that are classified as benign should perhaps be added to the benign table for a longer period of time. A potential avenue for future work would be to ban IP addresses for longer periods (e.g. greater than 24 hours) every successive time they are banned from the network. Implementing an increasing ban length based on ban count could potentially lower the effect serial spammers have on the network, as every time they attack the network they incur a longer ban. A similar implementation could also be applied to benign IPs, with the length of time they stay on the benign table set longer for successive benign classifications.

The results detailed above indicate that a system that looks three days back in the amalgamated audit logs offers the best compromise between timeliness, accuracy and the effectiveness of the system. Even though looking back over a whole month's worth of activity provided the system with the greatest accuracy, a look back time period of three days means the system would see the full activity spectrum of approximately 90% of all IPs, while operating at a significantly greater level of efficiency. By looking back for a time period of greater than 24 hours, the system is still able to retain full banning and blocking capabilities whilst maintaining good accuracy for benign IPs. Also, if the network administrator was prepared to accept lower accuracy when detection benign IPs or was able to implement a different method of benign IP tracking, a log look back time of 24 hours could be used.

6 Conclusions and Future Work

The research presented in this thesis focused on developing techniques to identify and catch spam at the network gateway. The ultimate aim of the project was to reduce the strain caused by spam on the network's internal infrastructure, in particular the mail servers as they no longer have to process as many spam messages. Lowering the load on the network's mail servers leaves them with more time to perform their primary duty of forwarding legitimate emails to their intended recipients.

The initial goal of this study was to validate the Honours work started by MacLeavy in 2004. Specifically, it needed to be determined if the detection of spam precursors could be used to create a system running in real-time that could identify the imminent arrival of spam and block it at the network gateway. Analysis of gateway audit log files identified potential spam precursor activity, but unfortunately this activity was found in such low proportions that it was deemed unsuitable for use in a real-time spam prevention system.

With network precursor detection deemed unsuitable, it was then proposed that spam messages themselves could be used as precursors, allowing the system to identify the current IP addresses of spammers and block them from accessing the network. In order to provide protection against legitimate emails being blocked by the system, a system of IP suspicion was implemented, with IP addresses being classified as malicious or benign based on the collection of supporting evidence.

Analysis of the timeliness of the system led to the conclusion that it executed with sufficient efficiency to make real-time operation viable. The speed of the system is directly linked to the amount of time that is looked back in the audit logs when searching for precursor spam activity. This amount of time also influences the effectiveness of the system, as accuracy is lowered when the log look back amount is

too small. It was determined that a look back amount of three days provided a good balance between system timeliness and effectiveness.

6.1 Future Work

The research detailed in this thesis has shown the viability of a spam prevention system at the network gateway, providing multiple avenues into further research within the area of spam protection using intrusion detection system techniques. The first area that needs to be investigated is to test the system using real spam filter log files instead of the simulated files used in this thesis. This would provide a true validation of the usefulness of the system and more definite statistics could be produced regarding the system's accuracy and effectiveness. An extension to this would be to run the system live upon a network with multiple gateways and mail servers. This would provide the most accurate form of validation, as the system's effect on overall network performance could be measured and compared to the gains in performance brought by the extra spam protection.

The amount of time that the system looks back in audit logs to gather evidence against IP addresses could be determined dynamically instead of being set at a static value. Log look back could increase in periods of lower network activity, as the system can afford to spend more processing power to increase the accuracy of IP classification. When network activity is higher than normal, the system could reduce the look back length to ensure network operation speed is not negatively affected while still maintaining some form of spam protection.

Another avenue for further work is the validation of the different suspicion and threshold values of the system. These values were chosen arbitrarily, based in part on the fact that simulated spam log files were used in this research. These values could be tested further and either validated as appropriate or alternative values could be found. Depending on the spam filter used, these values could be dynamically assigned based on how certain the spam filter is that the email is spam. In the case of SpamAssassin, this could be based on the number of rules the email matches while in the case of a Bayesian filter system it could be based on the spam probability

calculation of the email. Also, investigation could be carried out into the feasibility of dynamic threshold value adjustment, based on current system load.

Further research could also be undertaken on the current incremental suspicion system. An investigation could be carried out into the feasibility of using mathematical probabilities for the classification of IPs instead of arbitrary suspicion values. This could increase the accuracy of the system and provide extra assurance that legitimate emails are not being blocked by the system. To do this a thorough investigation would need to be carried out into the distinct proportions of spam email contained within the live audit logs in order to determine the probability that an IP is malicious or benign in intent.

The static ban length and benign classification length as set in the current system can also be examined. The figures in Table 5.5 indicate that IP addresses classified as benign should stay on the Benign IP table for a longer time period than the current system value of 24 hours. Investigation into the appropriate value for this time period has the potential to improve both the accuracy and the timeliness of the system. These time periods could also be set according to each individual IPs suspicion value at the time of classification. The larger the suspicion value, the longer amount of time the IP stays banned from the system of on the Benign IP table. The number of times the IP has been classified as malicious or benign could also influence the ban or benign length.

Outbound mail logs could be analysed to provide further insight into determining if an IP is malicious or benign. If an email address from the network has sent messages to other mail exchanges, there is a high likelihood that mail received from those exchanges is legitimate. This would essentially be adding support for email communication sessions, as mail received by the external exchange could be left unfiltered by the system for a period of time after communication from inside the network has commenced.

Email precursors found in Phase 1, although deemed not useful in the context of identifying the impending arrival of spam, may be useful in detecting the IP addresses of zombie machines compromised by Internet worms. If a unique activity

pattern, such as sending port scans and then sending spam can be used to identify compromised machines, this information could be used to protect the network from further attack and could also be used to inform the administrators of other networks about potential zombie activity.

In conclusion, this research has shown that it is viable to build a real-time system using audit log analysis techniques to stop spam at the network gateway. The use of amalgamated audit logs from multiple gateways has also shown that the entire network can be protected in the same way. Furthermore, this thesis has created a platform for future research within the area of spam detection and its prevention at the network gateway.

7 References

Allman, E 2003, 'Spam, Spam, Spam, Spam, Spam, the FTC, and Spam', *Queue*, vol. 1, no. 6, pp. 62-9.

Amoroso, EG 1999, *Intrusion detection : an introduction to Internet surveillance, correlation, traps, trace back, and response*, 1st edn, Intrusion.Net Books, Sparta, N.J.

The Apache SpamAssassin Project, 2005, viewed July 19 2005, <http://spamassassin.apache.org/index.html>.

Baranowski, S 2003, *How Secure are the Root DNS Servers?*, SANS Institute.

Barracuda Networks (Date Unknown), *An Overview of Spam Blocking Techniques*, Barracuda Networks, viewed Aug 22 2005, <http://www.barracudanetworks.com/ns/downloads/barracuda_spam_blocking_techniques.pdf>.

Bekker, S 2003, *Spam to Cost U.S. Companies $10 Billion in 2003*, ENT News, viewed May 11 2005, <http://www.entmag.com/news/article.asp?EditorialsID=5651>.

Costales, B and Flynt, M 2005, *Sendmail Milters: A Guide for Fighting Spam*, Pearson Education Inc.

Cranor, LF and LaMacchia, BA 1998, 'Spam!' *Commun. ACM*, vol. 41, no. 8, pp. 74-83.

Damiani, E, Vimercati, SDCd, Paraboschi, S and Samarati, P 2004, 'An Open Digest-based Technique for Spam Detection', *The 2004 International Workshop on Security in Parallel and Distributed Systems*, San Francisco, CA USA.

Danisch, H 2004, *Work in Progress, Internet-Draft: The RMX DNS RR and method for lightweight SMTP sender authorization*, Internet Engineering Task Force, viewed May 5 2005, <http://www.danisch.de/work/security/txt/draft-danisch-dns-rr-smtp-04.txt>.

Delany, M 2005, *Work in Progress, Internet-Draft: Domain-based Email Authentication Using Public-Keys Advertised in the DNS (DomainKeys)*, Internet Engineering Task Force, viewed Nov 29 2005, <http://www.ietf.org/internet-drafts/draft-delany-domainkeys-base-03.txt>.

Dougherty, C and Householder, A 2002, *CERT® Incident Note IN-2002-04*, CERT Coordination Center, viewed Aug 22 2005, <http://www.cert.org/incident_notes/IN-2002-04.html>.

Dougherty, C, Havrilla, J, Hernan, S and Lindner, M 2003, *CERT® Advisory CA-2003-20 W32/Blaster worm*, CERT Coordination Center, viewed Aug 22 2005, <http://www.cert.org/advisories/CA-2003-20.html>.

Drewes, R 2002, *An artificial neural network spam classifier*, Rich Drewes, viewed May 8 2005, <http://www.interstice.com/drewes/cs676/spam-nn/spam-nn.html>.

Fallows, D 2003, *Spam. How it is Hurting Email and Degrading Life on the Internet*, Pew Internet & American Life Project.

Fecyk, G 2003, *Work in Progress, Internet-Draft: Designated Mailers Protocol*, Internet Engineering Task Force, viewed May 5 2005, <http://www.pan-am.ca/dmp/draft-fecyk-dmp-01.txt>.

Garcia, FD, Hoepman, J-H and van Nieuwenhuizen, J 2004, 'Spam Filter Analysis', *19th IFIP International Information Security Conference*, Toulouse, France.

Gauthronet, S and Drouard, É 2001, *Unsolicited Commercial Communications and Data Protection*, Commission of the European Communities.

Golbeck, J and Hendler, J 2004, 'Reputation Network Analysis for Email Filtering', *Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, July 2004.

Grabnar, M 2004, *File :: Tail - Perl extension for reading from continously updated files*, viewed Aug 24 2005, <http://search.cpan.org/~mgrabnar/File-Tail-0.99.1/Tail.pm>.

Graham, P 2003, 'Better Bayesian Filtering', *2003 Spam Conference*.

Grote, M 2004, *An Overview of the Sender Policy Framework*, MS Exchange, viewed May 5 2005, <http://www.msexchange.org/tutorials/Sender-Policy-Framework.html>.

Haskins, R and Nielsen, D 2005, *Slamming Spam*, Pearson Education, Inc.

Hulten, G, Goodman, J and Rounthwaite, R 2004, 'Filtering spam e-mail on a global scale', in *Proceedings of the 13th international World Wide Web conference on Alternate track papers \& posters*, ACM Press, New York, NY, USA, pp. 366-7.

Jones, S 2003, *Port 0 OS Fingerprinting*, Network Penetration, viewed Aug 22 2005, <http://www.networkpenetration.com/port0.html>.

Jung, J and Sit, E 2004, 'An empirical study of spam traffic and the use of DNS black lists', in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, ACM Press, Taormina, Sicily, Italy, pp. 370-5.

Jung, J, Paxson, V, Berger, AW and Balakrishnan, H 2004, 'Fast Portscan Detection Using Sequential Hypothesis Testing', *IEEE Symposium on Security and Privacy*, Oakland, California, USA, 9-12 May.

Kemmerer, RA and Vigna, G 2002, 'Intrusion detection: a brief history and overview', *Computer*, vol. 35, no. 4, pp. 27-30.

LeMay, R 2005, *Spam sees Westnet blocked by BigPond*, ZDNet Australia, viewed Aug 15 2005, <http://www.zdnet.com.au/news/communications/soa/Spam_sees_Westnet_blocked_by_BigPond/0,2000061791,39204739,00.htm>.

Levy, E 2003, 'The making of a spam zombie army. Dissecting the Sobig worms', *Security & Privacy Magazine, IEEE*, vol. 1, no. 4, pp. 58-9.

Lonvick, C 2001, *RFC 3164: The BSD Syslog Protocol*, Network Working Group, viewed May 9 2005, <http://www.faqs.org/rfcs/rfc3164.html>.

Lueg, CP 2004, 'The Hidden Impacts of Anti-Spam Measures and their Contribution to the Digital Divide: An Exploratory Study', *The Annual Meeting of the American Society for Information Science and Technology (ASIS&T)*, Providence/Rhode Island, November 13-18, 2004.

Lyon, J 2004, *Work in Progress, Internet-Draft: Purported Responsible Address in E-Mail Messages*, Internet Engineering Task Force, viewed May 6 2005, <http://www.ietf.org/internet-drafts/draft-lyon-senderid-pra-00.txt>.

Lyon, J and Wong, M 2004, *Work in Progress, Internet-Draft: Sender ID: Authenticating E-Mail*, Internet Engineering Task Force, viewed May 6 2004, <http://download.microsoft.com/download/6/c/5/6c53077f-013e-480c-a19d-787850d84861/senderid_spec1.pdf>.

MacLeavy, C 2004, 'Predicting Spam Attacks based upon Detecting Preliminary Activity from Multiple Gateway Logs', Honours thesis, University of Tasmania.

Manderson, K 2005, to D Cook.

MessageLabs 2005, *MessageLabs Email Threats - Overview*, MessageLabs, viewed Aug 15 2005, <http://www.messagelabs.co.uk/publishedcontent/publish/threat_watch_dotcom_en/threat_statistics/spam_intercepts/DA_114633.chp.html>.

Microsoft Corporation 2005, *Sender ID Technology: Information for IT Professionals*, Microsoft Corporation, viewed May 6 2005, <http://www.microsoft.com/mscorp/safety/technologies/senderid/technology.mspx>.

National Office for the Information Economy 2003, *SPAM: Final Report of the NOIE Review of the Spam Problem and How It Can Be Solved*, National Office for the Information Economy.

O'Brien, C and Vogel, C 2003, 'Spam filters: bayes vs. chi-squared; letters vs. words', in *Proceedings of the 1st international symposium on Information and communication technologies*, Trinity College Dublin, Dublin, Ireland, pp. 291-6.

Olsen, S 2003, *AT&T spam filter loses valid e-mail*, viewed May 5 2005, <http://news.com.com/ATT+spam+filter+loses+valid+e-mail/2100-1023_3-982118.html>.

Pantel, P and Lin, D 1998, 'SpamCop: A Spam Classication & Organization Program', *AAAI-98 Workshop on Learning for Text Categorization*.

Paulson, LD 2004, 'Spam hits instant messaging', *Computer*, vol. 37, no. 4, p. 18.

Pfleeger, SL and Bloom, G 2005, 'Canning Spam: Proposed Solutions to Unwanted Email', *Security & Privacy Magazine, IEEE*, vol. 3, no. 2, pp. 40-7.

Sahami, M, Dumais, S, Heckerman, D and Horvitz, E 1998, 'A Bayesian Approach to Filtering Junk E-Mail', *AAAI-98 Workshop on Learning for Text Categorization*.

Seifried, K 2003, *Information security / TCP Ports list, UDP ports list*, viewed Aug 24 2005, <http://www.seifried.org/security/ports/>.

Spam Prevention Early Warning System, 2005, SPEWS.org, viewed 17 Aug 2005, <http://www.spews.org/>.

The Spamhaus Project, 2005, The Spamhaus Project Ltd, viewed Aug 17 2005, <http://www.spamhaus.org/>.

Stoll, C 1991, *The cuckoo's egg : tracking a spy through the maze of computer espionage*, Pan Books, London.

Trend Micro RBL+ Service, 2005, Trend Micro Incorporated, viewed Aug 17 2005, <http://www.trendmicro.com/en/products/nrs/rbl/evaluate/overview.htm>.

Wagner, J 2004, *Sender ID Still Making Tracks*, Inside ID, viewed May 6 2005, <http://www.insideid.com/id_management/article.php/3409601>.

Wald, A 1947, *Sequential Analysis*, John Wiley and Sons, New York.

Weiss, A 2003, 'Ending spam's free ride', *netWorker*, vol. 7, no. 2, pp. 18-24.

Whitworth, B and Whitworth, E 2004, 'Spam and the social-technical gap', *Computer*, vol. 37, no. 10, pp. 38-45.

Wong, M and Schlitt, W 2004, *Work in Progress, Internet-Draft: Sender Policy Framework: Authorizing Use of Domains in E-MAIL*, Internet Engineering Task Force, viewed May 5 2005, <http://ietf.org/internet-drafts/draft-schlitt-spf-classic-00.txt>.

Yoshida, K, Adachi, F, Washio, T, Motoda, H, Homma, T, Nakashima, A, Fujikawa, H and Yamazaki, K 2004, 'Density-based spam detector', in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, Seattle, WA, USA, pp. 486-93.